Ultra-Low Power 8-bit MCU, Flash memory 16KB, SRAM 1KB, 12-bit ADC, RTCC, and 128-Bit Unique ID

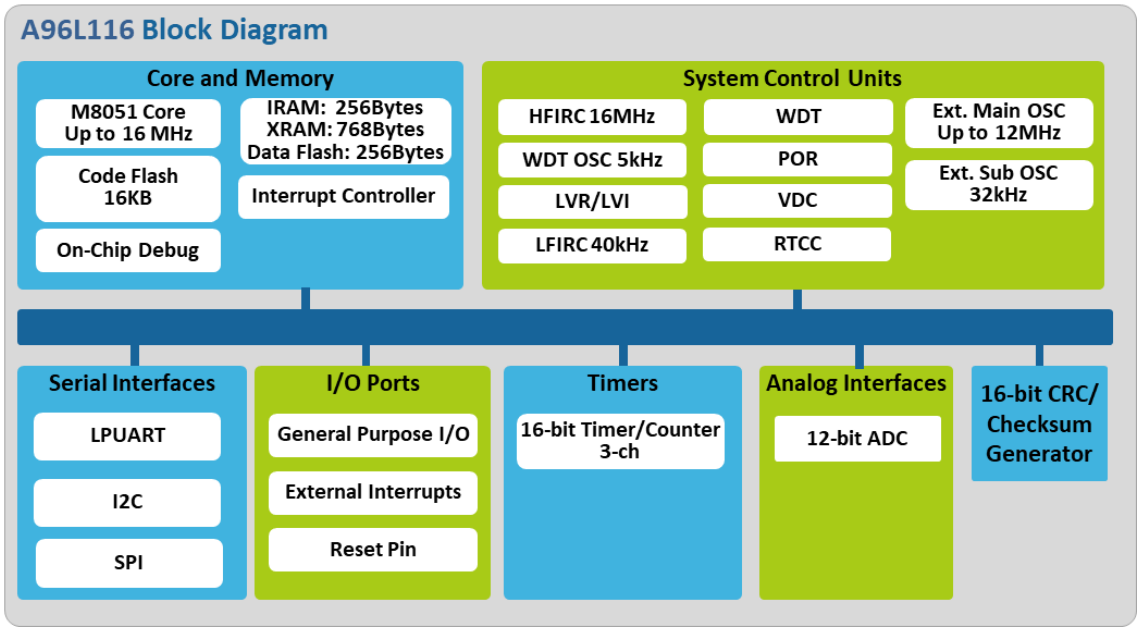
UM Rev. 1.02

Introduction

This user's manual contains complete information for application developers who use A96L116 for their specific needs.

The A96L116 is an advanced CMOS 8-bit microcontroller for ultra-low power applications. It has 16 Kbytes of code flash memory, 256 bytes of data flash memory, and peripherals. It provides a highly flexible and cost-effective solution for many embedded applications. A96L116 supports power-down modes, reducing power consumption.

Figure 1. A96L116 Block Diagram



Reference Documents

- A96L116 Datasheet includes information on mechanical characteristics, development methods, and ordering information.
  It is available at the ABOV website: https://www.abovsemi.com/.
- SDK 51 User's guide (System Design Kit): Intel released it in 1982. It contains all the components of a single-board computer based on Intel's 8051 single-chip microcomputer.
- Information on Mentor Graphics 8051 microcontroller: This technical document is provided at Mentor® website, https://www.mentor.com/products/ip/peripheral/microcontroller/

Contents

List of Tables

List of Figures

1 Description

A96L116 is an advanced CMOS ultra-low power 8-bit microcontroller. It has 16 Kbytes of flash memory, 256 bytes data flash memory, 256 bytes of IRAM, 768 bytes of XRAM, general-purpose I/O, basic interval timer, watchdog timer, 16-bit timer/counter, 16-bit PPG output, real-time clock/calendar, LPUART, SPI, I2C, 12-bit A/D converter, on-chip POR, LVR, LVI, on-chip oscillator and clock circuitry. A96L116 supports power-down modes, reducing power consumption.

1.1 Device Overview

Table 1 describes features of A96L116 and peripheral counts.

Table 1. A96L116 Device Features and Peripheral Counts

Peripheral		A96L116
CPU		• 8-bit M8051 core, two clocks per cycle
Flash Memory	Code Flash	• 16 Kbytes with self-read/write capability • On-chip debug and ISP • Endurance: 10,000 cycles
	Data Flash	• 256 bytes • Endurance: 100,000 cycles
RAM	IRAM	• 256 bytes
	XRAM	• 768 bytes
GPIO		• Normal I/O • 18 ports: P0[7:0], P1[5:0], P2[3:0]
Timer/Counter		• BIT 8-bit x 1-ch • WDT 8-bit x 1-ch: 5 kHz internal RC oscillator for WDT • 16-bit x 3-ch (T0/T1/T2) • Real-time clock/calendar (RTCC)
Programmable Pulse Generation		• Pulse generation (by T0/T1/T2)
ADC		• 12-bit ADC, eight input channels
CRC and Checksum Generator		• 16-bit • Auto and user CRC/Checksum mode
Reset	Power-On Reset	• Reset release level (1.2V)
	Low-Voltage Reset	• 8 Level detect (1.5 V to 2.78 V)
Low-Voltage Indicator		• 7 Level detect (1.87 V to 2.78 V)
Seral Interface		• Low power UART, up to 9,600 bps with 32.768 kHz • I2C x 1, SPI x 1

Table 1. A96L116 Device Features and Peripheral Counts (continued)

Peripheral	A96L116
High Frequency Internal RC oscillator	• 16 MHz ±2.0% (TA = −40°C to 85°C)
Low Frequency Internal RC oscillator	• 40 kHz ±15% (TA = −40°C to 85°C)
Power Consumption	• 94 µA/MHz in Run mode, • 12 µA in Run mode (32.768 kHz, 40 kHz) • 0.35 µA in Stop mode without RTCC • 0.9 µA in Stop mode with RTCC and 32.768 kHz • 5 µs wakeup time from idle/stop modes
Operating voltage and frequency	• 1.71 V to 3.6 V @ 32 to 38 kHz with SX-tal • 1.8 V to 3.6 V @ 0.4 to 4.2 MHz with Ceramic • 2.0 V to 3.6 V @ 0.4 to 4.2 MHz with Crystal • 2.7 V to 3.6 V @ 0.4 to 12 MHz with X-tal • 1.71 V to 3.6 V @ 0.5 to 16 MHz with HFIRC • 1.71 V to 3.6 V @ 40 kHz with LFIRC
Minimum instruction execution time	• 0.125 µs @ 16 MHz IRC
Operating temperature	• −40°C to 85°C
Package type	• 20-TSSOP (0.65 mm pitch) • 20-QFN (0.5 mm pitch) • 16-SOPN (1.27 mm pitch) • 10-SSOP (1.0 mm pitch) • Pb-free package

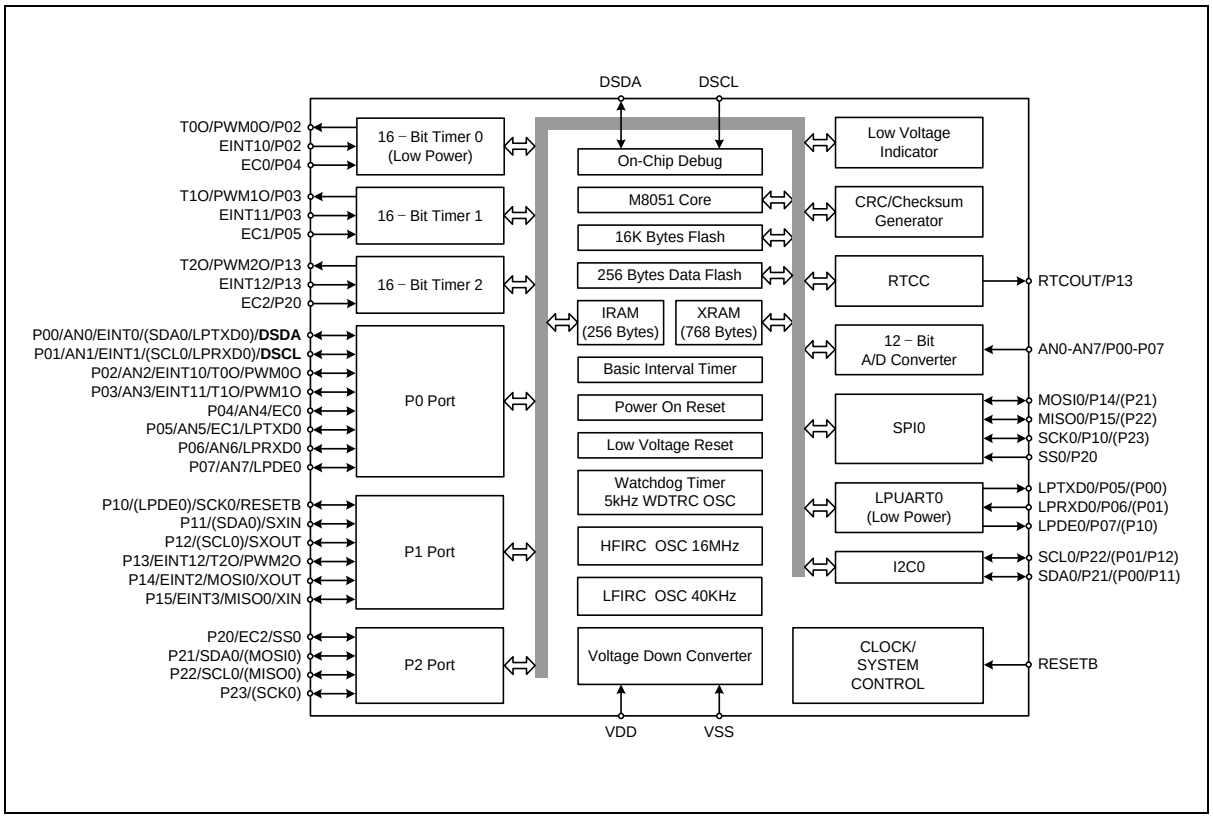
Table 2. Summary of A96L116 Peripherals

Peripheral		A96L116FR	A96L116FU	A96L116AE	A96L116DS
Code Flash Memory		16 KB	16 KB	16 KB	16 KB
Data Flash Memory		256 B	256 B	256 B	256 B
IRAM/XRAM		256/768 B	256/768 B	256/768 B	256/768 B
Timers	General-Purpose	3 (16-bit)			
	WDT	1			
	BIT	1			
Communication Interfaces	SPI	1	1	1	0
	I2C	1	1	1	1
	LPUART	1	1	1	1
RTCC		1			
GPIO		18	18	14	8
ADCs		111.1 ksps	111.1 ksps	111.1 ksps	111.1 ksps
Number of channels		8	8	7	4
Max. Operating Frequency		16 MHz			
128-bit Unique ID		1			
CRC/Checksum		1 (16-bit)			
Operating Voltage		1.71 V to 3.6 V			
Operating Temperature		Ambient operating temperature: −40°C to 85°C			
Packages		20-TSSOP	20-QFN	16-SOPN	10-SSOP

1.1.1 Block Diagram

Figure 2 describes A96L116 in a block diagram.

Figure 2. A96L116 Block Diagram

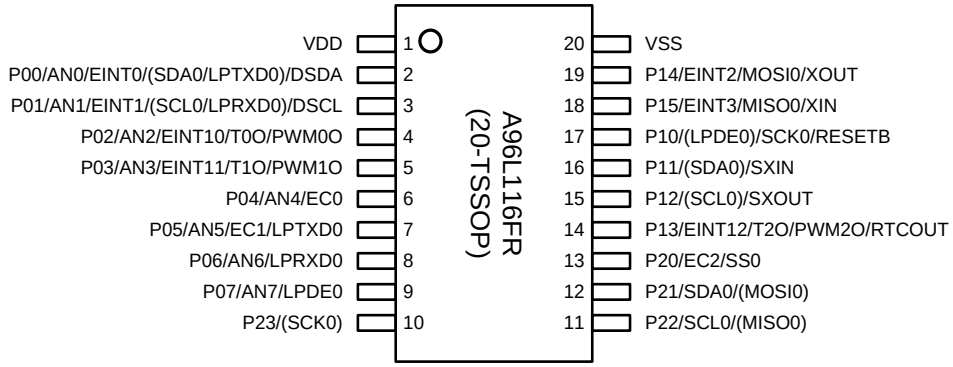


2 Pinouts and Pin Descriptions

This chapter describes the A96L116 pinouts and pin descriptions.

2.1 Pinouts

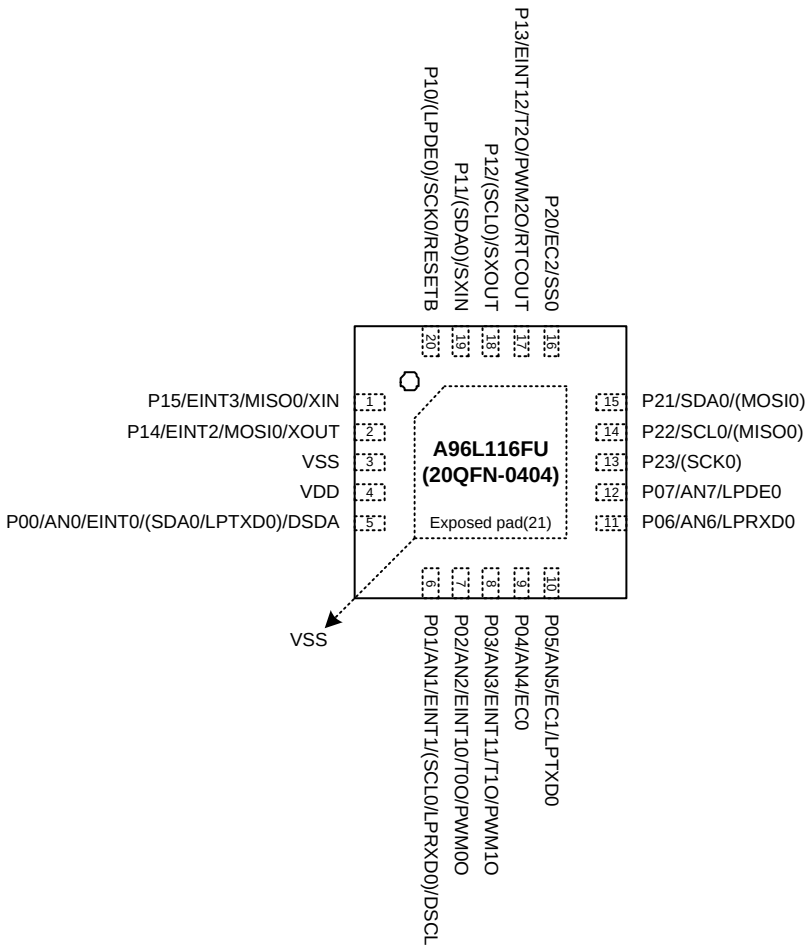
Figure 3. A96L116FR 20-TSSOP Pinouts



NOTE:

1. For the On-Chip Debugging, ISP assigns P0[1:0] pins for DSCL and DSDA, respectively.

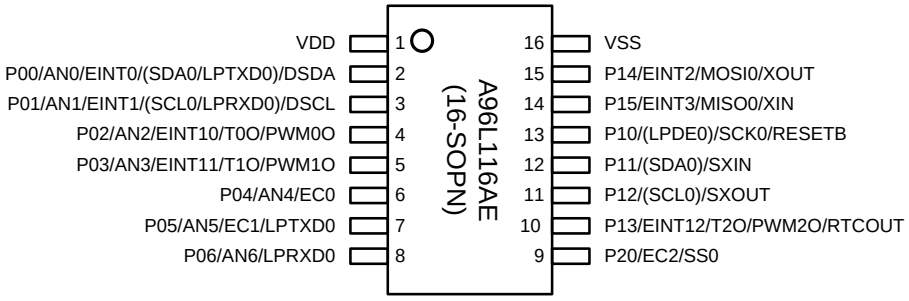
Figure 4. A96L116FU 20-QFN Pinouts



NOTE:

1. For the On-Chip Debugging, ISP assigns P0[1:0] pins for DSCL and DSDA, respectively.

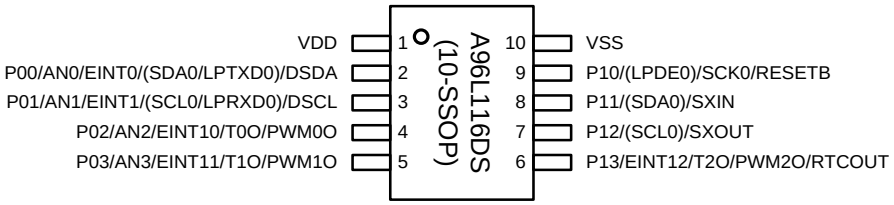
Figure 5. A96L116AE 16-SOPN Pinouts



NOTES:

1. For the On-Chip Debugging, ISP assigns P0[1:0] pins for DSCL and DSDA, respectively.
2. The P07 and P2[3:1] pins should be selected as a push-pull output or an input with a pull-up resistor by software control when the 16-pin package is used

Figure 6. A96L116DS 10-SSOP Pinouts



NOTES:

1. For the On-Chip Debugging, ISP assigns P0[1:0] pins for DSCL and DSDA, respectively.
2. The P0[7:4], P1[5:4], and P2[3:0] pins should be selected as a push-pull output or an input with a pull-up resistor by software control when the 10-pin package is used.

2.2 Pin Description

Table 3. Pin Description

Pin Name	I/O	Function	@reset	Shared with
P00	I/O	The port 0 is a bit-programmable I/O port that can be configured as a Schmitt-trigger input, a push-pull output, or an open-drain output. A pull-up resistor can be specified in a 1-bit unit.	Input	AN0/EINT0/(SDA0/LPTXD0)/DSDA
P01				AN1/EINT1/(SCL0/LPRXD0)/DSCL
P02				AN2/EINT10/T0O/PWM0O
P03				AN3/EINT11/T1O/PWM1O
P04				AN4/EC0
P05				AN5/EC1/LPTXD0
P06				AN6/LPRXD0
P07				AN7/LPDE0
P10	I/O	Port 1 is a bit-programmable I/O port that can be configured as a Schmitt-trigger input, a push-pull output, or an open-drain output. A pull-up resistor can be specified in a 1-bit unit.	Input	(LPDE0)/SCK0/RESETB
P11				(SDA0)/SXIN
P12				(SCL0)/SXOUT
P13				EINT12/T2O/PWM2O/RTCOUT
P14				EINT2/MOSI0/XOUT
P15				EINT3/MISO0/XIN
P20	I/O	Port 2 is a bit-programmable I/O port that can be configured as a Schmitt-trigger input, a push-pull output, or an open-drain output. A pull-up resistor can be specified in a 1-bit unit.	Input	EC2/SS0
P21				SDA/(MOSI0)
P22				SCL/(MISO0)
P23				(SCK0)
EINT0	I/O	External interrupt inputs	Input	P00/AN0/(SDA0/LPTXD0)/DSDA
EINT1				P01/AN1/(SCL0/LPRXD0)/DSCL
EINT2				P14/MOSI0/XOUT
EINT3				P15/MISO0/XIN
EINT10	I/O	External interrupt input and Timer 0 capture input	Input	P02/AN2/T0O/PWM0O
EINT11		External interrupt input and Timer 1 capture input		P03/AN3/T1O/PWM1O
EINT12		External interrupt input and Timer 2 capture input		P13/T2O/PWM2O

Table 3. Pin Description (continued)

Pin Name	I/O	Function	@reset	Shared with
T0O	I/O	Timer 0 interval output	Input	P02/AN2/EINT10/PWM0O
T1O	I/O	Timer 1 interval output	Input	P03/AN3/EINT11/PWM1O
T2O	I/O	Timer 2 interval output	Input	P13/EINT12/PWM2O
PWM0O	I/O	Timer 0 pulse output	Input	P02/AN2/EINT10/T0O
PWM1O	I/O	Timer 1 pulse output	Input	P03/AN3/EINT11/T1O
PWM2O	I/O	Timer 2 pulse output	Input	P13/EINT12/T2O
EC0	I/O	Timer 0 event count input	Input	P04/AN4
EC1	I/O	Timer 1 event count input	Input	P05/AN5/LPTXD0
EC2	I/O	Timer 2 event count input	Input	P20/SS0
AN0	I/O	A/D converter analog input channels	Input	P00/EINT0/(SDA0/LPTXD0)/DSDA
AN1				P01/EINT1/(SCL0/LPRXD0)/DSCL
AN2				P02/EINT10/T0O/PWM0O
AN3				P03/EINT11/T1O/PWM1O
AN4				P04/EC0
AN5				P05/EC1/LPTXD0
AN6				P06/LPRXD0
AN7				P07/LPDE0
LPTXD0	I/O	Low power UART data output	Input	P05/AN5/EC1 (P00/AN0/EINT0/(SDA0)/DSDA)
LPRXD0	I/O	Low power UART data input	Input	P06/AN6 (P01/AN1/EINT1/(SCL0)/DSCL)
LPDE0	I/O	Low power UART DE signal output	Input	P07/AN7 (P10/SCK0/RESETB)
MOSI0	I/O	SPI master output, slave input	Input	P14/EINT2/XOUT (P21/SDA0)
MISO0	I/O	SPI master input, slave output	Input	P15/EINT3/XIN (P22/SCL0)
SCK0	I/O	SPI clock input/output	Input	P10/(LPDE0)/RESETB (P23)
SS0	I/O	SPI slave select input	Input	P20/EC2

Table 3. Pin Description (continued)

Pin Name	I/O	Function	@reset	Shared with
SCL0	I/O	I2C clock input/output	Input	P22/(MISO0) (P01/AN1/EINT1/LPRXD0/DSCL) (P12/SXOUT)
SDA0	I/O	I2C data input/output	Input	P21/(MOSI0) (P00/AN0/EINT0/LPTXD0/DSDA) (P11/SXIN)
RTCOUT	I/O	Real-time clock output	Input	P13/EINT12/T2O/PWM2O
RESETB	I/O	System reset pin with a pull-up resistor when it is selected as the RESETB by “CONFIGURE OPTION”	Input	P10/(LPDE0)/SCK0
DSDA	I/O	On-chip debugger data input/output	Input	P00/AN0/EINT0/(SDA0/LPTXD0)
DSCL	I/O	On-chip debugger clock input	Input	P01/AN1/EINT1/(SCL0/LPRXD0)
XIN	I/O	Main oscillator pins	Input	P15/EINT3/MISO0
XOUT				P14/EINT2/MOSI0
SXIN	I/O	Sub oscillator pins	Input	P11/(SDA0)
SXOUT				P12/(SCL0)
VDD, VSS	–	Power input pins	–	–

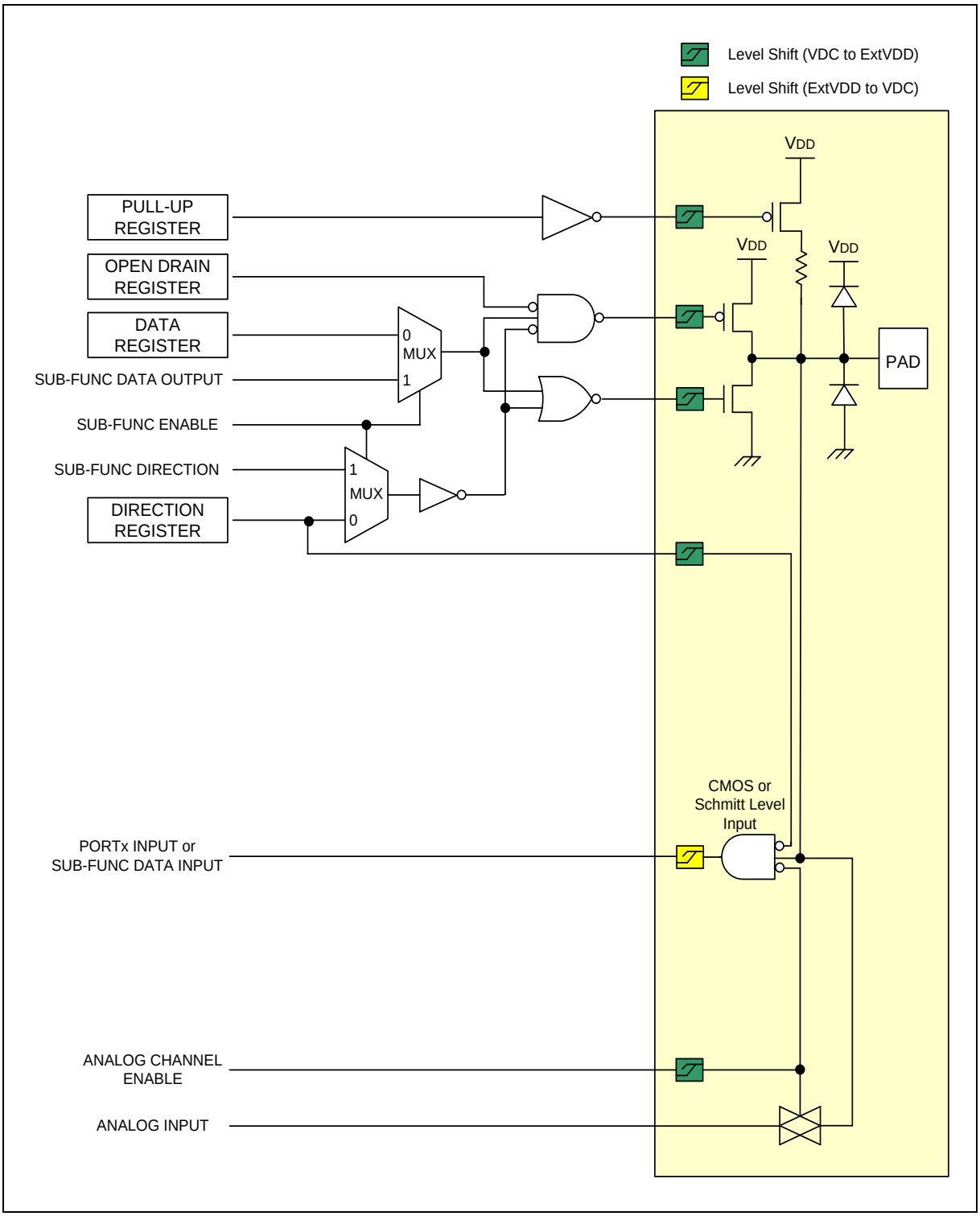
NOTES:

1. The P10/RESETB pin is configured as one of the P10 and RESETB pins by the “CONFIGURE OPTION”.
2. The P15/XIN and P14/XOUT pins are configured as function pins by software control.
3. The P11/SXIN and P12/SXOUT pins are configured as function pins by s/w control.
4. If the P00/DSDA and P01/DSCL pins are connected to an emulator during Power-On Reset, the pins are automatically configured as the debugger pins.
5. The P00/DSDA and P01/DSCL pins are configured as inputs with internal pull-up resistors only during the reset or Power-On Reset.

3 Port Structures

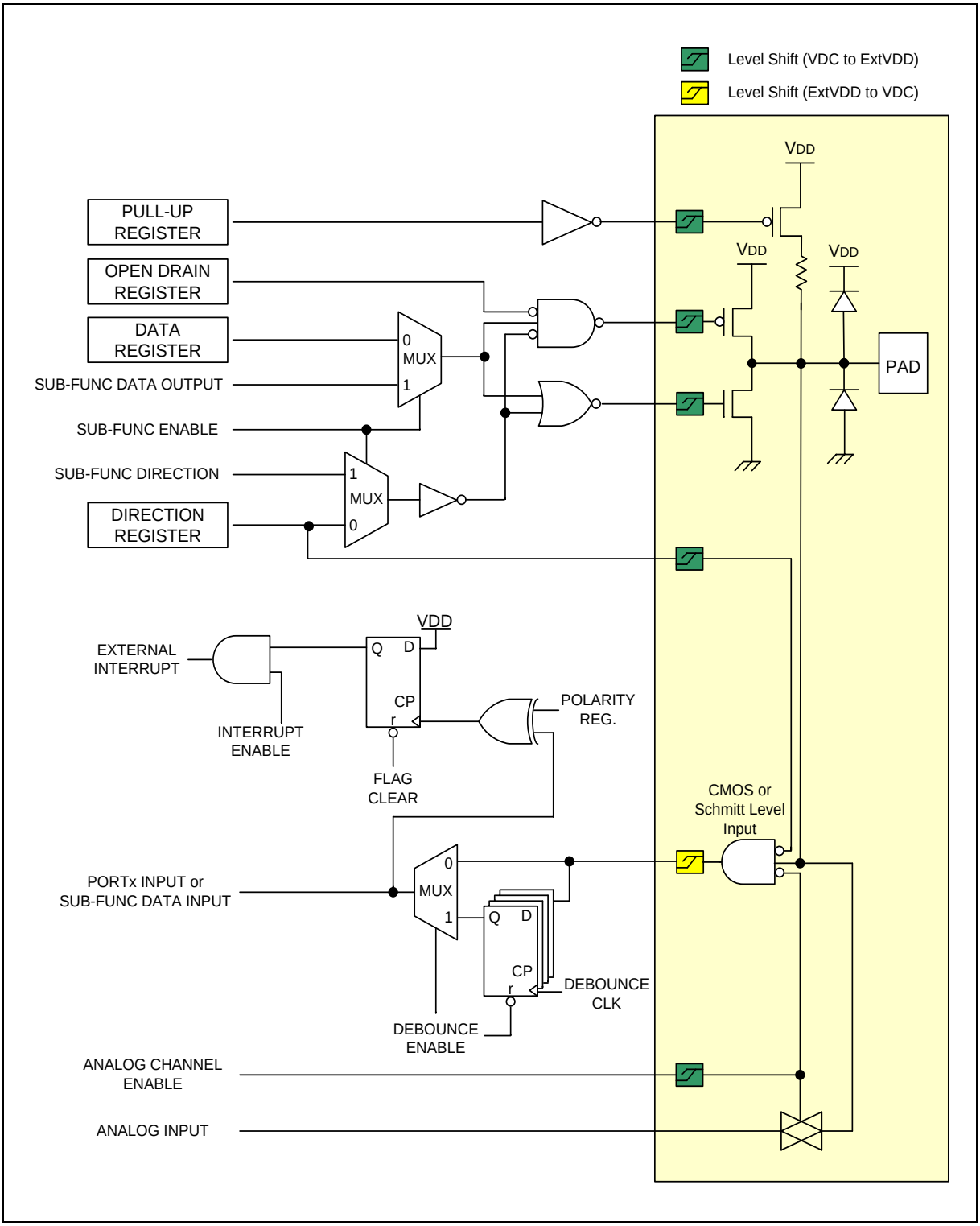
3.1 GPIO Port Structure

Figure 7. General-Purpose I/O Port Structure



3.2 External Interrupt I/O Port Structure

Figure 8. External Interrupt I/O Port Structure



4 Memory Organization

The A96L116 addresses three separate memory spaces:

- Program memory
- Data memory
- XRAM memory

An 8-bit CPU address can rapidly access the data memory through this logical memory separation. 16-bit data memory address is generated through the DPTR register.

A96L116 provides on-chip 16 Kbytes of ISP type flash program memory, which is readable and writable. Internal data memory (IRAM) is 256 bytes, including the stack area. External data memory (XRAM) is 768 bytes.

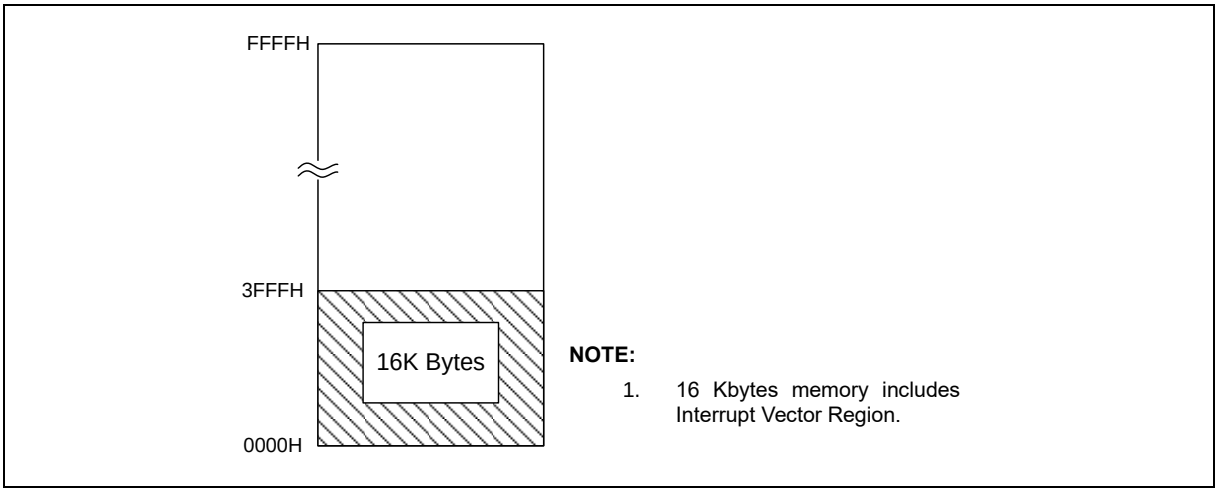
4.1 Program Memory

A 16-bit program counter can address up to 64 Kbytes, but the A96L116 has only 16 Kbytes program memory space. After reset, the CPU begins execution from location 0000H. Each interrupt is assigned to a fixed location of the program memory. The interrupt causes the CPU to jump to that location, where it commences an execution of a service routine.

For example, an external interrupt 1 is assigned to location 002BH. If the external interrupt 1 is going to be used, its service routine must begin at location 002BH. If the interrupt is not going to be used, its service location is available as general-purpose program memory. If an interrupt service routine is short enough (frequent cases with a control application), the service routine can reside entirely within an 8-byte interval.

A longer service routine can use a jump instruction to skip over subsequent interrupt locations if other interrupts are in use. Figure 9 shows a map of the lower part of the program memory.

Figure 9. Program Memory



A more detailed description of program memory is described in chapter 20, later in this document.

4.2 Internal Data Memory

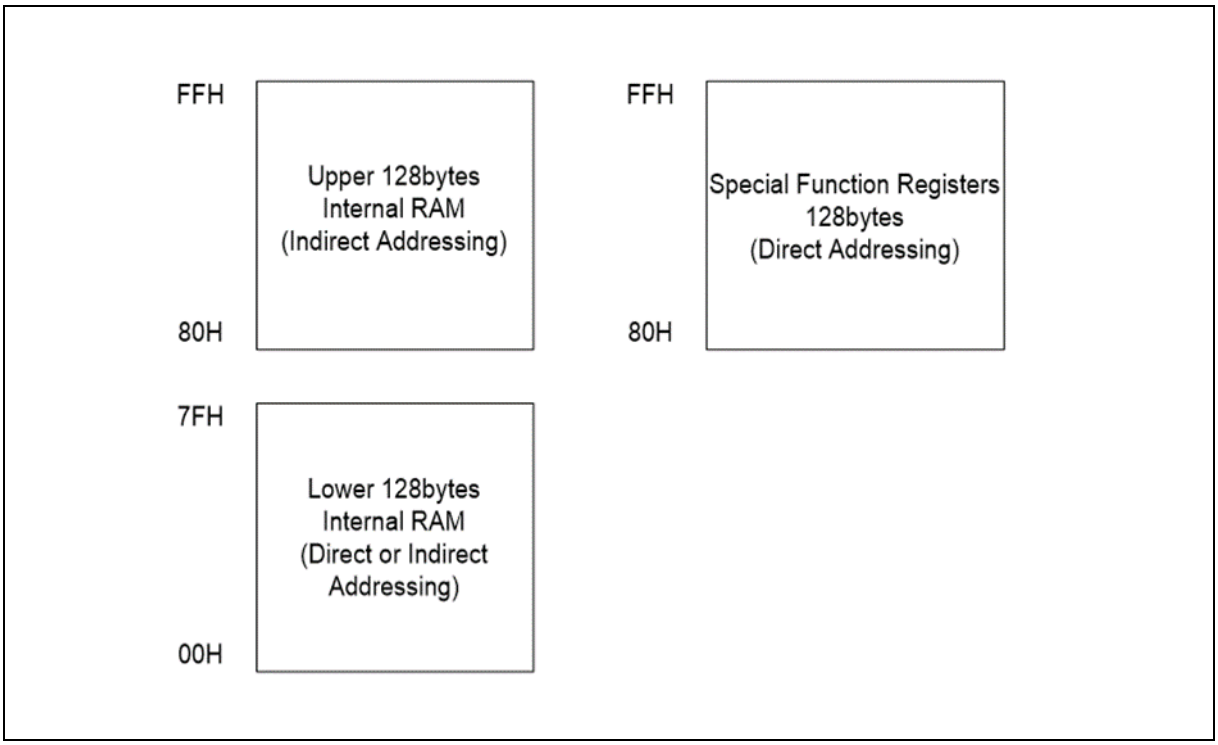
Internal data memory is divided into three spaces, as shown in Figure 10. Those three spaces are generally called as,

- Lower 128 bytes
- Upper 128 bytes
- Special Function Registers (SFR space)

Internal data memory addresses are always one byte wide, which implies an address space of 256 bytes.

The internal data memory addressing modes can accommodate up to 384 bytes by using a simple trick. Direct addresses higher than 7FH access one memory space, and indirect addresses higher than 7FH access a different memory space. This method allows the upper 128 bytes and SFR space to occupy the same block of addresses, 80H through FFH, although they are physically separate entities, as shown in Figure 10.

Figure 10. Internal Data Memory Map

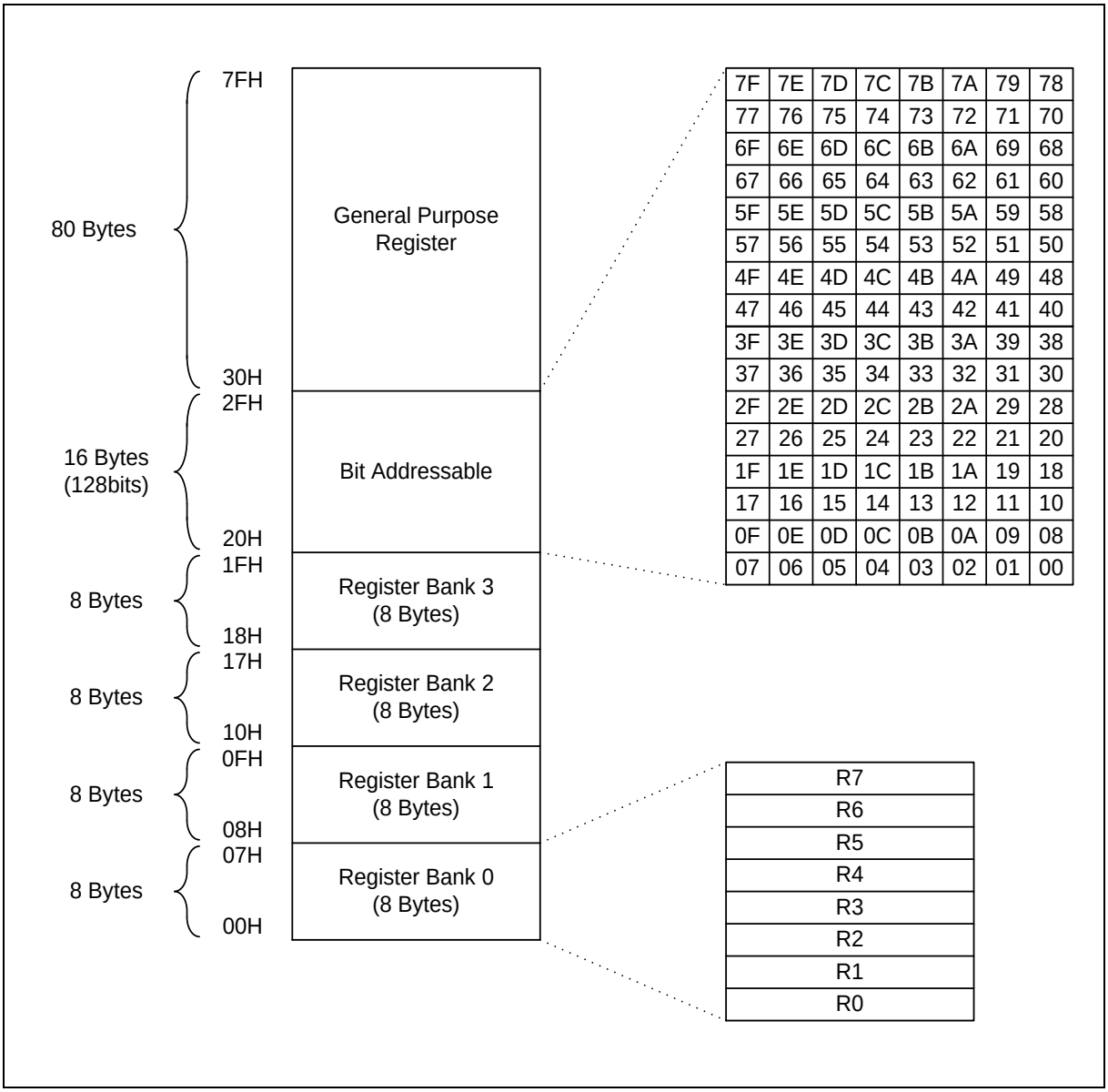


The lower 128 bytes of RAM are present in all 8051 devices, as mapped in Figure 11. The lowest 32 bytes are grouped into four banks of 8 registers. Program instructions call out these registers as R0 through R7. Two bits in the Program Status Word select which register bank is in use. This allows more efficient use of code space since register instructions are shorter than direct addressing instructions.

The next 16 bytes above the register banks form a block of bit-addressable memory space. The 8051 instructions set includes a wide selection of single-bit instructions, which can directly address the 128 bits in this area. The bit addresses in this area are 00H through 7FH.

Entire bytes in the lower 128 bytes can be accessed by direct or indirect addressing, while the upper 128 bytes RAM can only be accessed by indirect addressing. These spaces are used for data RAM and stack.

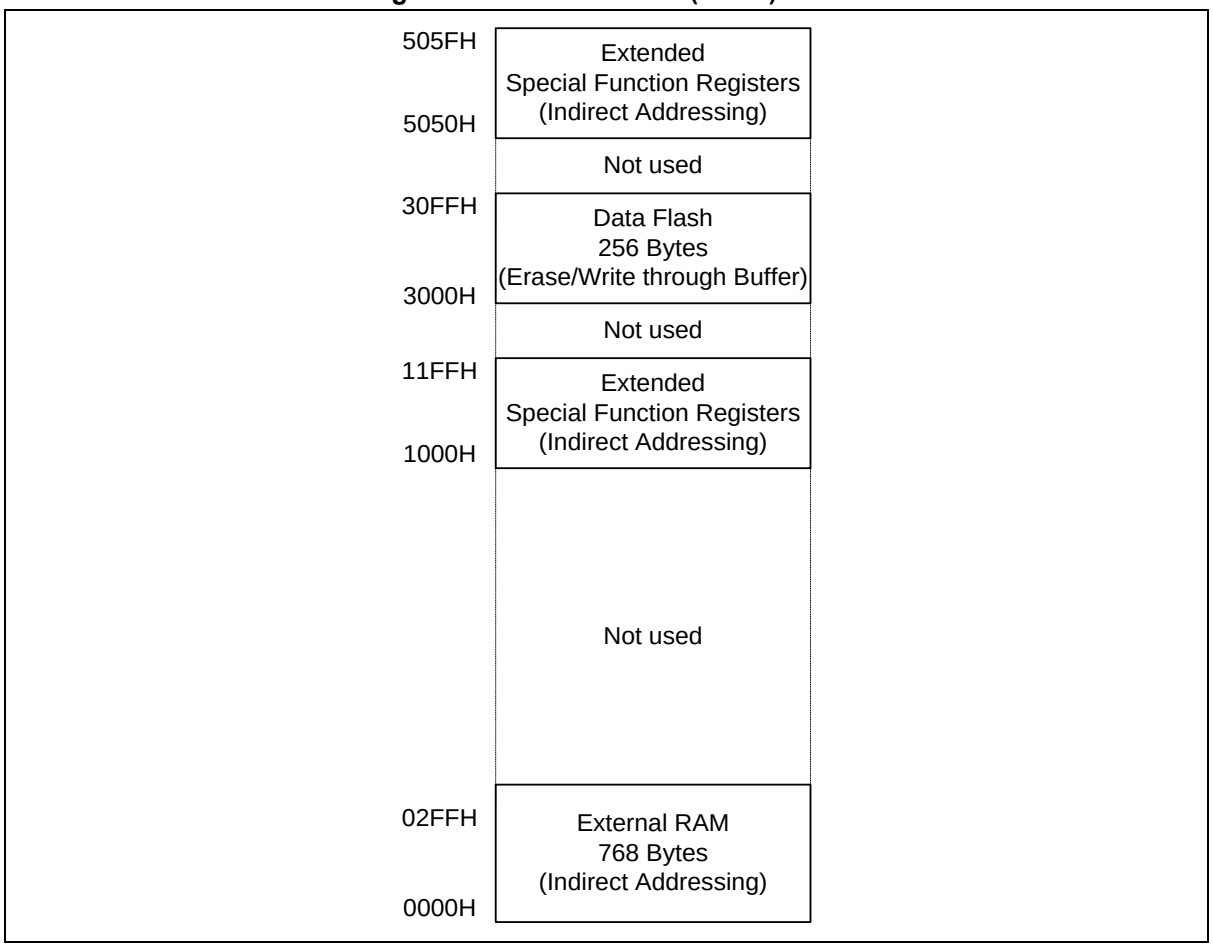
Figure 11. Lower 128 bytes Internal RAM



4.3 Extended SFR and Data Memory Area

The A96L116 has 256-byte XRAM and XSFR registers. The extended SFR area has no relation with RAM or flash. This area can be read or written using SFR in an 8-bit unit.

Figure 12. Extended SFR (XSFR) Area

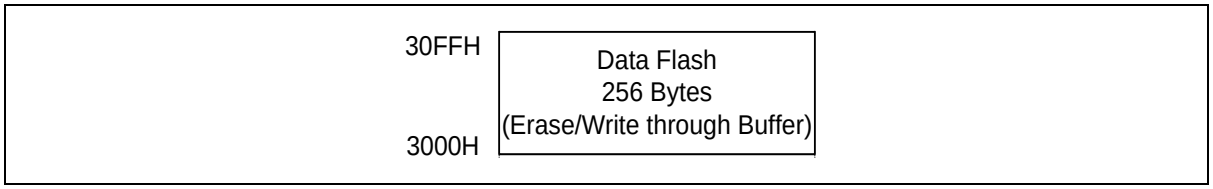


4.4 Data Flash Area

Data flash area has no relation with RAM or flash. This area can be read by using DPTR. The data flash area can be erased or written by using a buffer.

For detailed information about the Data Flash, please refer to Chapter 21.

Figure 13. Data Flash Area



4.5 SFR Map

This section describes the information on the SFR map and map summaries in Table 4 through Table 7.

4.5.1 SFR Map Summary

Table 4. SFR Map Summary

	00H/8H[NOTE]	01H/9H	02H/0AH	03H/0BH	04H/0CH	05H/0DH	06H/0EH	07H/0FH
0F8H	IP1	–	FSADRH	FSADRM	FSADRL	FIDR	FMCR	–
0F0H	B	MPWRCR	DFSADRL	DFSADRH	DFIDR	DFMCR	I2C0SAR0	I2C0SAR1
0E8H	RSTFR	PPCLKEN3	I2C0CR	I2C0SR	I2C0DR	I2C0SDHR	I2C0SCLR	I2C0SCHR
0E0H	ACC	PPCLKEN2	LPUT0CR0	LPUT0CR1	LPUT0CR2	LPUT0CR3	LPUT0CR4	LPUT0IER
0D8H	LVRCR	PPCLKEN1	LPUT0ISRL	LPUT0ISRH	LPUT0RDR	LPUT0TDR	LPUT0BDR	–
0D0H	PSW	PPCLKEN0	LPUT0BCPL	LPUT0BCPH	LPUT0RTDRL	LPUT0RTDRH	LPUT0RCDR	LPUT0DLY
0C8H	OSCCR	SUBISET	T0CRL	T0CRH	T0ADRL	T0ADRH	T0BDRL	T0BDRH
0C0H	–	–	P2IOL	–	P2OD	P2PU	P2FSRL	–
0B8H	IP	P1DB	P1IOL	P1IOH	P1OD	P1PU	P1FSRL	P1FSRH
0B0H	–	P0DB	P0IOL	P0IOH	P0OD	P0PU	P0FSRL	P0FSRH
0A8H	IE	IE1	IE2	IE3	EIPOL2L	–	–	–
0A0H	–	EIFLAG2	EO	–	–	–	–	–
98H	–	EIFLAG1	EIPOL0L	–	SPI0CR	SPI0DR	SPI0SR	IRCIDR
90H	P2	EIFLAG0	ADCCRL	ADCCRH	ADCDRL	ADCDRH	–	–
88H	P1	FCDIN	SCCR	BITCR	BITCNT	WDTCR	WDTDR/ WDTCNT	IRCTCR
80H	P0	SP	DPL	DPH	DPL1	DPH1	LVICR	PCON

– Reserved M8051 compatible

NOTE:

1. Registers 00H/8H are bit-addressable except OSCCR.

4.5.2 Extended SFR Map Summary

Table 5. XSFR Map Summary

	00H/8H	01H/9H	02H/0AH	03H/0BH	04H/0CH	05H/0DH	06H/0EH	07H/0FH
5058H	FCDRL	–	–	–	–	–	–	LVRIDR
5050H	FCSARH	FCEARH	FCSARM	FCEARM	FCSARL	FCEARL	FCCR	FCDRH
.....	–	–	–	–	–	–	–	–
10B8H	RTCDAY	RTCWK	RTCMTH	RTCYR	RTCAMIN	RTCAHR	RTCAWK	–
10B0H	RTCCRL	RTCCRH	RTCSCTL	RTCSCTH	RTCECR	RTCSEC	RTCMIN	RTCHR
.....	–	–	–	–	–	–	–	–
1068H	UNIQUEID8	UNIQUEID9	UNIQUEID10	UNIQUEID11	UNIQUEID12	UNIQUEID13	UNIQUEID14	UNIQUEID15
1060H	UNIQUEID0	UNIQUEID1	UNIQUEID2	UNIQUEID3	UNIQUEID4	UNIQUEID5	UNIQUEID6	UNIQUEID7
.....	–	–	–	–	–	–	–	–
1010H	T2CRL	T2CRH	T2ADRL	T2ADRH	T2BDRL	T2BDRH	T2CAPL	T2CAPH
1008H	T1CRL	T1CRH	T1ADRL	T1ADRH	T1BDRL	T1BDRH	T1CAPL	T1CAPH
1000H	IRCCTRM	IRCFTRM	–	–	–	-	T0CAPL	T0CAPH

– Reserved

4.5.3 SFR Map

Table 6. SFR Map

Address	Function	Symbol	R/W	@ Reset							
				7	6	5	4	3	2	1	0
80H	P0 Data Register	P0	R/W	0	0	0	0	0	0	0	0
81H	Stack Pointer	SP	R/W	0	0	0	0	0	1	1	1
82H	Data Pointer Register Low	DPL	R/W	0	0	0	0	0	0	0	0
83H	Data Pointer Register High	DPH	R/W	0	0	0	0	0	0	0	0
84H	Data Pointer Register Low 1	DPL1	R/W	0	0	0	0	0	0	0	0
85H	Data Pointer Register High 1	DPH1	R/W	0	0	0	0	0	0	0	0
86H	Low-Voltage Indicator Control Register	LVICR	R/W	–	–	0	0	–	0	0	0
87H	Power Control Register	PCON	R/W	–	–	–	–	–	–	0	0
88H	P1 Data Register	P1	R/W	–	0	0	0	0	0	0	0
89H	Flash CRC Data In Register	FCDIN	R/W	0	0	0	0	0	0	0	0
8AH	System and Clock Control Register	SCCR	R/W	–	–	–	–	–	–	0	0
8BH	BIT Control Register	BITCR	R/W	0	0	0	–	0	0	0	1
8CH	Basic Interval Timer Counter Register	BITCNT	R	0	0	0	0	0	0	0	0
8DH	Watchdog Timer Control Register	WDTCR	R/W	0	0	0	–	–	0	0	0
8EH	Watchdog Timer Data Register	WDTDR	W	1	1	1	1	1	1	1	1
8EH	Watchdog Timer Counter Register	WDTCNT	R	0	0	0	0	0	0	0	0
8FH	Internal RC Trim Control Register	IRCTCR	R/W	0	0	0	0	0	0	0	0
90H	P2 Data Register	P2	R/W	–	–	–	–	0	0	0	0
91H	External Interrupt Flag 0 Register	EIFLAG0	R/W	–	–	–	–	0	0	0	0
92H	A/D Converter Control Low Register	ADCCRL	R/W	0	0	–	0	0	0	0	0
93H	A/D Converter Control High Register	ADCCRH	R/W	0	–	0	0	0	0	0	0
94H	A/D Converter Data Low Register	ADCDRL	R	x	x	x	x	x	x	x	x
95H	A/D Converter Data High Register	ADCDRH	R	x	x	x	x	x	x	x	x
96H	Reserved	–	–	–							
97H	Reserved	–	–	–							
98H	Reserved	–	–	–							
99H	External Interrupt Flag 1 Register	EIFLAG1	R/W	0	–	–	–	–	–	–	–
9AH	External Interrupt Polarity 0 Low Register	EIPOL0L	R/W	0	0	0	0	0	0	0	0

Table 6. SFR Map (continued)

Address	Function	Symbol	R/W	@ Reset							
				7	6	5	4	3	2	1	0
9BH	Reserved	–	–	–							
9CH	SPI0 Control Register	SPI0CR	R/W	0	0	0	0	0	0	0	0
9DH	SPI0 Data Register	SPI0DR	R/W	0	0	0	0	0	0	0	0
9EH	SPI0 Status Register	SPI0SR	R/W	0	0	0	–	0	0	–	–
9FH	Internal RC Trim Identification Register	IRCIDR	R/W	0	0	0	0	0	0	0	0
A0H	Reserved	–	–	–							
A1H	External Interrupt Flag 2 Register	EIFLAG2	R/W	–	–	–	–	–	0	0	0
A2H	Extended Operation Register	EO	R/W	–	–	–	0	–	0	0	0
A3H	Reserved	–	–	–							
A4H	Reserved	–	–	–							
A5H	Reserved	–	–	–							
A6H	Reserved	–	–	–							
A7H	Reserved	–	–	–							
A8H	Interrupt Enable Register	IE	R/W	0	–	0	–	–	0	0	0
A9H	Interrupt Enable Register 1	IE1	R/W	–	–	–	–	–	0	0	0
AAH	Interrupt Enable Register 2	IE2	R/W	–	–	–	–	–	0	0	0
ABH	Interrupt Enable Register 3	IE3	R/W	–	–	–	0	0	0	0	0
ACH	External Interrupt Polarity 2 Low Register	EIPOL2L	R/W	–	–	0	0	0	0	0	0
ADH	Reserved	–	–	–							
AEH	Reserved	–	–	–							
AFH	Reserved	–	–	–							
B0H	Reserved	–	–	–							
B1H	P0 Debounce Enable Register	P0DB	R/W	0	0	–	–	0	0	0	0
B2H	P0 Direction Low Register	P0IOL	R/W	1	1	1	1	0	0	0	0
B3H	P0 Direction High Register	P0IOH	R/W	1	1	1	1	1	1	1	1
B4H	P0 Open-drain Selection Register	P0OD	R/W	0	0	0	0	0	0	0	0
B5H	P0 Pull-up Resistor Selection Register	P0PU	R/W	0	0	0	0	0	0	0	0
B6H	Port 0 Function Selection Low Register	P0FSRL	R/W	–	0	–	0	0	0	0	0

Table 6. SFR Map (continued)

Address	Function	Symbol	R/W	@ Reset							
				7	6	5	4	3	2	1	0
B7H	Port 0 Function Selection High Register	P0FSRH	R/W	–	0	–	0	0	0	–	0
B8H	Interrupt Priority Register	IP	R/W	–	–	0	0	0	0	0	0
B9H	P1 Debounce Enable Register	P1DB	R/W	–	–	0	0	0	–	–	–
BAH	P1 Direction Low Register	P1IOL	R/W	0	0	0	0	0	0	1	1
BBH	P1 Direction High Register	P1IOH	R/W	–	–	–	–	1	1	1	1
BCH	P1 Open-drain Selection Register	P1OD	R/W	–	–	0	0	0	0	0	0
BDH	P1 Pull-up Resistor Selection Register	P1PU	R/W	–	–	0	0	0	0	0	0
BEH	Port 1 Function Selection Low Register	P1FSRL	R/W	–	0	–	0	–	0	–	0
BFH	Port 1 Function Selection High Register	P1FSRH	R/W	–	–	–	–	–	0	–	0
C0H	Reserved	–	–	–							
C1H	Reserved	–	–	–							
C2H	P2 Direction Low Register	P2IOL	R/W	1	1	1	1	1	1	1	1
C3H	Reserved	–	–	–							
C4H	P2 Open-drain Selection Register	P2OD	R/W	–	–	–	–	0	0	0	0
C5H	P2 Pull-up Resistor Selection Register	P2PU	R/W	–	–	–	–	0	0	0	0
C6H	Port 2 Function Selection Low Register	P2FSRL	R/W	–	0	–	0	–	0	–	0
C7H	Reserved	–	–	–							
C8H	Oscillator Control Register	OSCCR	R/W	0	–	0	0	1	0	0	0
C9H	Sub Oscillator Driving Current Selection Register	SUBISET	R/W	–	–	–	–	–	1	1	1
CAH	Timer 0 Control Low Register	T0CRL	R/W	0	0	0	0	0	0	0	0
CBH	Timer 0 Control High Register	T0CRH	R/W	0	–	0	0	0	0	0	0
CCH	Timer 0 A Data Low Register	T0ADRL	R/W	1	1	1	1	1	1	1	1
CDH	Timer 0 A Data High Register	T0ADRH	R/W	1	1	1	1	1	1	1	1
CEH	Timer 0 B Data Low Register	T0BDRL	R/W	1	1	1	1	1	1	1	1
CFH	Timer 0 B Data High Register	T0BDRH	R/W	1	1	1	1	1	1	1	1
D0H	Program Status Word Register	PSW	R/W	0	0	0	0	0	0	0	0
D1H	Peripheral Clock Control Register 0	PPCLKEN0	R/W	0	0	–	–	–	0	0	0

Table 6. SFR Map (continued)

Address	Function	Symbol	R/W	@ Reset							
				7	6	5	4	3	2	1	0
D2H	LPUART0 Baud Rate Compensation Low Register	LPUT0BCPL	R/W	0	0	0	0	0	0	0	0
D3H	LPUART0 Baud Rate Compensation High Register	LPUT0BCPH	R/W	0	–	–	–	–	–	–	0
D4H	LPUART0 Receive Time Out Data Low Register	LPUT0RTDRL	R/W	1	1	1	1	1	1	1	1
D5H	LPUART0 Receive Time Out Data High Register	LPUT0RTDRH	R/W	1	1	1	1	1	1	1	1
D6H	LPUART0 Receive Character Detection Data Register	LPUT0RCDR	R/W	0	0	0	0	0	0	0	0
D7H	LPUART0 Tx Delay Time Data Register	LPUT0DLY	R/W	0	0	0	0	0	0	0	0
D8H	Low-Voltage Reset Control Register	LVRCR	R/W	0	–	–	–	0	0	0	0
D9H	Peripheral Clock Control Register 1	PPCLKEN1	R/W	0	0	–	–	–	0	0	0
DAH	LPUART0 Interrupt Flag and Status Low Register	LPUT0ISRL	R/W	–	–	–	0	0	0	1	0
DBH	LPUART0 Interrupt Flag and Status High Register	LPUT0ISRH	R/W	–	–	–	–	0	0	0	0
DCH	LPUART0 Receive Data Register	LPUT0RDR	R	0	0	0	0	0	0	0	0
DDH	LPUART0 Transmit Data Register	LPUT0TDR	R/W	0	0	0	0	0	0	0	0
DEH	LPUART0 Baud Rate Data Register	LPUT0BDR	R/W	1	1	1	1	1	1	1	1
DFH	Reserved	–	–	–							
E0H	Accumulator Register	ACC	R/W	0	0	0	0	0	0	0	0
E1H	Peripheral Clock Control Register 2	PPCLKEN2	R/W	0	–	–	–	0	–	–	0
E2H	LPUART0 Control Register 0	LPUT0CR0	R/W	0	0	0	0	–	0	0	0
E3H	LPUART0 Control Register 1	LPUT0CR1	R/W	–	0	0	0	0	0	0	0
E4H	LPUART0 Control Register 2	LPUT0CR2	R/W	–	–	–	0	0	0	0	0
E5H	LPUART0 Control Register 3	LPUT0CR3	R/W	–	–	–	0	0	0	0	0
E6H	LPUART0 Control Register 4	LPUT0CR4	R/W	–	–	–	0	0	–	0	0

Table 6. SFR Map (continued)

Address	Function	Symbol	R/W	@ Reset							
				7	6	5	4	3	2	1	0
E7H	LPUART0 Interrupt Enable Register	LPUT0IER	R/W	–	–	–	0	0	0	0	0
E8H	Reset Flag Register	RSTFR	R/W	1	x	0	0	x	–	–	–
E9H	Peripheral Clock Control Register 3	PPCLKEN3	R/W	0	0	–	0	–	–	–	0
EAH	I2C0 Control Register	I2C0CR	R/W	0	0	0	0	0	0	0	0
EBH	I2C0 Status Register	I2C0SR	R/W	0	0	0	0	0	0	0	0
ECH	I2C0 Data Register	I2C0DR	R/W	0	0	0	0	0	0	0	0
EDH	I2C0 SDA Hold Time Register	I2C0SDHR	R/W	0	0	0	0	0	0	0	1
EEH	I2C0 SCL Low Period Register	I2C0SCLR	R/W	0	0	1	1	1	1	1	1
EFH	I2C0 SCL High Period Register	I2C0SCHR	R/W	0	0	1	1	1	1	1	1
F0H	B Register	B	R/W	0	0	0	0	0	0	0	0
F1H	Memory Power Control Register	MPWRCR	R/W	–	–	–	0	–	–	–	0
F2H	Data Flash Sector Address Low Register	DFSADRL	R/W	0	0	0	–	–	–	–	–
F3H	Data Flash Sector Address High Register	DFSADRH	R/W	0	0	0	0	0	0	0	0
F4H	Data Flash Identification Register	DFIDR	R/W	0	0	0	0	0	0	0	0
F5H	Data Flash Mode Control Register	DFMCR	R/W	0	–	–	–	–	0	0	0
F6H	I2C0 Slave Address 0 Register	I2C0SAR0	R/W	0	0	0	0	0	0	0	0
F7H	I2C0 Slave Address 1 Register	I2C0SAR1	R/W	0	0	0	0	0	0	0	0
F8H	Interrupt Priority Register 1	IP1	R/W	–	–	0	0	0	0	0	0
F9H	Reserved	–	–	–							
FAH	Flash Sector Address High Register	FSADRH	R/W	–	–	–	–	0	0	0	0
FBH	Flash Sector Address Middle Register	FSADRM	R/W	0	0	0	0	0	0	0	0
FCH	Flash Sector Address Low Register	FSADRL	R/W	0	0	0	0	0	0	0	0
FDH	Flash Identification Register	FIDR	R/W	0	0	0	0	0	0	0	0
FEH	Flash Mode Control Register	FMCR	R/W	0	–	–	–	–	0	0	0
FFH	Reserved	–	–	–							

4.5.4 Extended SFR Map

Table 7. XSFR Map

Address	Function	Symbol	R/W	@ Reset							
				7	6	5	4	3	2	1	0
1000H	Internal RC Coarse Trim Register	IRCCTRM	R/W	–	–	–	–	–	0	0	0
1001H	Internal RC Fine Trim Register	IRCFTRM	R/W	–	–	0	0	0	0	0	0
1002H	Reserved	–	–	–							
1003H	Reserved	–	–	–							
1004H	Reserved	–	–	–							
1005H	Reserved	–	–	–							
1006H	Timer 0 Capture Data Low Register	T0CAPL	R	0	0	0	0	0	0	0	0
1007H	Timer 0 Capture Data High Register	T0CAPH	R	0	0	0	0	0	0	0	0
1008H	Timer 1 Control Low Register	T1CRL	R/W	0	0	0	0	0	0	0	0
1009H	Timer 1 Control High Register	T1CRH	R/W	0	–	0	0	–	–	0	0
100AH	Timer 1 A Data Low Register	T1ADRL	R/W	1	1	1	1	1	1	1	1
100BH	Timer 1 A Data High Register	T1ADRH	R/W	1	1	1	1	1	1	1	1
100CH	Timer 1 B Data Low Register	T1BDRL	R/W	1	1	1	1	1	1	1	1
100DH	Timer 1 B Data High Register	T1BDRH	R/W	1	1	1	1	1	1	1	1
100EH	Timer 1 Capture Data Low Register	T1CAPL	R	0	0	0	0	0	0	0	0
100FH	Timer 1 Capture Data High Register	T1CAPH	R	0	0	0	0	0	0	0	0
1010H	Timer 2 Control Low Register	T2CRL	R/W	0	0	0	0	0	0	0	0
1011H	Timer 2 Control High Register	T2CRH	R/W	0	–	0	0	–	–	0	0
1012H	Timer 2 A Data Low Register	T2ADRL	R/W	1	1	1	1	1	1	1	1
1013H	Timer 2 A Data High Register	T2ADRH	R/W	1	1	1	1	1	1	1	1
1014H	Timer 2 B Data Low Register	T2BDRL	R/W	1	1	1	1	1	1	1	1
1015H	Timer 2 B Data High Register	T2BDRH	R/W	1	1	1	1	1	1	1	1
1016H	Timer 2 Capture Data Low Register	T2CAPL	R	0	0	0	0	0	0	0	0
1017H	Timer 2 Capture Data High Register	T2CAPH	R	0	0	0	0	0	0	0	0

Table 7. XSFR Map(continued)

Address	Function	Symbol	R/W	@ Reset							
				7	6	5	4	3	2	1	0
1060H	Unique ID Register 0	UNIQUEID0	R	x	x	x	x	x	x	x	x
1061H	Unique ID Register 1	UNIQUEID1	R	x	x	x	x	x	x	x	x
1062H	Unique ID Register 2	UNIQUEID2	R	x	x	x	x	x	x	x	x
1063H	Unique ID Register 3	UNIQUEID3	R	x	x	x	x	x	x	x	x
1064H	Unique ID Register 4	UNIQUEID4	R	x	x	x	x	x	x	x	x
1065H	Unique ID Register 5	UNIQUEID5	R	x	x	x	x	x	x	x	x
1066H	Unique ID Register 6	UNIQUEID6	R	x	x	x	x	x	x	x	x
1067H	Unique ID Register 7	UNIQUEID7	R	x	x	x	x	x	x	x	x
1068H	Unique ID Register 8	UNIQUEID8	R	x	x	x	x	x	x	x	x
1069H	Unique ID Register 9	UNIQUEID9	R	x	x	x	x	x	x	x	x
106AH	Unique ID Register 10	UNIQUEID10	R	x	x	x	x	x	x	x	x
106BH	Unique ID Register 11	UNIQUEID11	R	x	x	x	x	x	x	x	x
106CH	Unique ID Register 12	UNIQUEID12	R	x	x	x	x	x	x	x	x
106DH	Unique ID Register 13	UNIQUEID13	R	x	x	x	x	x	x	x	x
106EH	Unique ID Register 14	UNIQUEID14	R	x	x	x	x	x	x	x	x
106FH	Unique ID Register 15	UNIQUEID15	R	x	x	x	x	x	x	x	x

Table 7. XSFR Map (continued)

Address	Function	Symbol	R/W	@ Reset							
				7	6	5	4	3	2	1	0
10B0H	RTCC Control Low Register	RTCCRL	R/W	0	0	0	–	0	0	0	0
10B1H	RTCC Control High Register	RTCCRH	R/W	0	0	0	0	0	0	–	0
10B2H	RTCC Sub-counter Low Register	RTCSCTL	R	0	0	0	0	0	0	0	0
10B3H	RTCC Sub-counter High Register	RTCSCTH	R	0	0	0	0	0	0	0	0
10B4H	RTCC Time Error Correction Register	RTCECR	R/W	0	0	0	0	0	0	0	0
10B5H	RTCC Second Counter Register	RTCSEC	R/W	–	0	0	0	0	0	0	0
10B6H	RTCC Minute Counter Register	RTCMIN	R/W	–	0	0	0	0	0	0	0
10B7H	RTCC Hour Counter Register	RTCHR	R/W	–	–	0	1	0	0	1	0
10B8H	RTCC Day Counter Register	RTCDAY	R/W	–	–	0	0	0	0	0	1
10B9H	RTCC Week Counter Register	RTCWK	R/W	–	–	–	–	–	0	0	0
10BAH	RTCC Month Counter Register	RTCMTH	R/W	–	–	–	0	0	0	0	1
10BBH	RTCC Year Counter Register	RTCYR	R/W	0	0	0	0	0	0	0	0
10BCH	RTCC Alarm Minute Register	RTCAMIN	R/W	–	0	0	0	0	0	0	0
10BDH	RTCC Alarm Hour Register	RTCAHR	R/W	–	–	0	1	0	0	1	0
10BEH	RTCC Alarm Week Register	RTCAWK	R/W	–	0	0	0	0	0	0	0

5050H	Flash CRC Start Address High Register	FCSARH	R/W	–	–	–	–	–	–	–	0
5051H	Flash CRC End Address High Register	FCEARH	R/W	–	–	–	–	–	–	–	0
5052H	Flash CRC Start Address Middle Register	FCSARM	R/W	0	0	0	0	0	0	0	0
5053H	Flash CRC End Address Middle Register	FCEARM	R/W	0	0	0	0	0	0	0	0
5054H	Flash CRC Start Address Low Register	FCSARL	R/W	0	0	0	0	–	–	–	–
5055H	Flash CRC End Address Low Register	FCEARL	R/W	0	0	0	0	–	–	–	–
5056H	Flash CRC Control Register	FCCR	R/W	0	0	0	–	0	0	0	0
5057H	Flash CRC Data High Register	FCDRH	R	1	1	1	1	1	1	1	1
5058H	Flash CRC Data Low Register	FCDRL	R	1	1	1	1	1	1	1	1

505FH	LVR Write Identification Register	LVRIDR	R/W	0	0	0	0	0	0	0	0

4.5.5 SFR Map

ACC (Accumulator Register): E0H

7	6	5	4	3	2	1	0
ACC							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

ACC	Accumulator

B (B Register): F0H

7	6	5	4	3	2	1	0
B							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

B	B Register

SP (Stack Pointer): 81H

7	6	5	4	3	2	1	0
SP							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 07H

SP	Stack Pointer

DPL (Data Pointer Register Low): 82H

7	6	5	4	3	2	1	0
DPL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

DPL	Data Pointer Low

DPH (Data Pointer Register High): 83H

7	6	5	4	3	2	1	0
DPH							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

DPH	Data Pointer High

DPL1 (Data Pointer Register Low 1): 84H

7	6	5	4	3	2	1	0
DPL1							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

DPL1	Data Pointer Low 1

DPH1 (Data Pointer Register High 1): 85H

7	6	5	4	3	2	1	0
DPH1							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

DPH1	Data Pointer High 1

PSW (Program Status Word Register): D0H

7	6	5	4	3	2	1	0
CY	AC	F0	RS1	RS0	OV	F1	P
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

CY	Carry Flag
AC	Auxiliary Carry Flag
F0	General-Purpose User-Definable Flag
RS1	Register Bank Select bit 1
RS0	Register Bank Select bit 0
OV	Overflow Flag
F1	User-Definable Flag
P	Parity Flag. Set/Cleared by hardware each instruction cycle to indicate an odd/even number of ‘1’ bits in the accumulator

EO (Extended Operation Register): A2H

7	6	5	4	3	2	1	0
–	–	–	TRAP_EN	–	DPSEL2	DPSEL1	DPSEL0
–	–	–	R/W	–	R/W	R/W	R/W

Initial value: 00H

TRAP_EN Select the Instruction **(Keep always ‘0’)**.

0 Select MOVC @(DPTR++), A

1 Select Software TRAP Instruction

DPSEL[2:0] Select Banked Data Pointer Register

DPSEL2	DPSEL1	SPSEL0	Description
0	0	0	DPTR0
0	0	1	DPTR1
Reserved			

5 Ports

5.1 I/O Ports

A96L116 has three groups of I/O ports, P0, P1, and P2. Each port can be easily configured as an input pin, an output, or an internal pull up and open-drain pin by software. The port configuration pursues to meet various system configurations and design requirements.

5.2 Port Registers

5.2.1 Data Register (Px)

The data register (Px) is related to a bidirectional I/O port. If a port is configured as an output port, data can be written to the corresponding bit of the Px. If a port is configured as an input, data can be read from the corresponding bit of the Px.

5.2.2 Direction Register (PxIOH/L)

The PxIOH/L register selects one from the input and output modes for each port pin. Each pin can be configured as an input pin, an output pin, or an Alternative Function pin.

Most bits are set to ‘1b’ by a system reset, but the system reset clears some bits.

5.2.3 Pull-up Register Selection Register (PxPU)

On-chip pull-up resistors can be connected to I/O ports individually by configuring a pull-up resistor selection register (PxPU). Setting a PxPU register can enable or disable a pull-up resistor of each port. If a certain bit in the PxPU register is 1, a pull-up resistor of the corresponding pin is enabled. While the bit is 0, the pull-up resistor is disabled. A system reset clears all bits.

5.2.4 Open-drain Selection Register (PxOD)

There are internal open-drain selection registers (PxOD) for Px. Setting a PxOD register can enable or disable an open-drain of each port. Most ports become push-pull by a system reset, but some ports become open-drain by the system reset.

5.2.5 Debounce Enable Register (P0DB, P1DB)

P00, P01, P02, P03, P13, P14, P15 support a debounce function. Debounce clocks of the ports are fx/1, fx/4, fx/16, and fx/64, respectively.

5.2.6 Port Function Selection Register (P0FSRH, P0FSRL, P1FSRH, P1FSRL, P2FSRL)

Port function selection registers define alternative functions of ports. Please remember that these registers must be set properly for alternative port functions.

5.2.7 Register Map

Table 8. Port Register Map

Name	Address	Direction	Default	Description
P0	80H	R/W	00H	P0 Data Register
P0IOH	B3H	R/W	FFH	P0 Direction High Register
P0IOL	B2H	R/W	F0H	P0 Direction Low Register
P0OD	B4H	R/W	00H	P0 Open-drain Selection Register
P0PU	B5H	R/W	00H	P0 Pull-up Resistor Selection Register
P0DB	B1H	R/W	00H	P0 Debounce Enable Register
P0FSRH	B7H	R/W	00H	P0 Function Selection High Register
P0FSRL	B6H	R/W	00H	P0 Function Selection Low Register
P1	88H	R/W	00H	P1 Data Register
P1IOH	BBH	R/W	0FH	P1 Direction High Register
P1IOL	BAH	R/W	03H	P1 Direction Low Register
P1OD	BCH	R/W	00H	P1 Open-drain Selection Register
P1PU	BDH	R/W	00H	P1 Pull-up Resistor Selection Register
P1DB	B9H	R/W	00H	P1 Debounce Enable Register
P1FSRH	BFH	R/W	00H	P1 Function Selection High Register
P1FSRL	BEH	R/W	00H	P1 Function Selection Low Register
P2	90H	R/W	00H	P2 Data Register
P2IOL	C2H	R/W	FFH	P2 Direction Low Register
P2OD	C4H	R/W	00H	P2 Open-drain Selection Register
P2PU	C5H	R/W	00H	P2 Pull-up Resistor Selection Register
P2FSRL	C6H	R/W	00H	P2 Function Selection Low Register

5.3 Port P0

5.3.1 Port Description of P0

As an 8-bit I/O port, P0 controls the following registers:

- P0 data register (P0)
- P0 direction registers (P0IOH/L)
- P0 debounce enable register (P0DB)
- P0 pull-up resistor selection register (P0PU)
- P0 open-drain selection register (P0OD)
- P0 Function selection registers (P0FSRH/P0FSRL)

5.3.2 Register Description of P0

P0 (P0 Data Register): 80H

7	6	5	4	3	2	1	0
P07	P06	P05	P04	P03	P02	P01	P00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

P0[7:0]	I/O Data

P0PU (P0 Pull-up Resistor Selection Register): B5H

7	6	5	4	3	2	1	0
P07PU	P06PU	P05PU	P04PU	P03PU	P02PU	P01PU	P00PU
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

P0PU[7:0]	Configure Pull-up Resistor of P0 Port	
	0	Disable
	1	Enable

P0OD (P0 Open-drain Selection Register): B4H

7	6	5	4	3	2	1	0
P07OD	P06OD	P05OD	P04OD	P03OD	P02OD	P01OD	P00OD
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

P0OD[7:0]	Configure Open-drain of P0 Port	
	0	Push-pull output
	1	Open-drain output

P0IOH (P0 Direction High Register): B3H

7	6	5	4	3	2	1	0
P0IOH7	P0IOH6	P0IOH5	P0IOH4	P0IOH3	P0IOH2	P0IOH1	P0IOH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

P0IOH[7:6] P07 Function select

P0IOH7	P0IOH6	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode(Both input and output are disabled)

P0IOH[5:4] P06 Function select

P0IOH5	P0IOH4	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode(Both input and output are disabled)

P0IOH[3:2] P05 Function select

P0IOH3	P0IOH2	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode(Both input and output are disabled)

P0IOH[1:0] P04 Function select

P0IOH1	P0IOH0	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode(Both input and output are disabled)

P0IOL (P0 Direction Low Register): B2H

7	6	5	4	3	2	1	0
P0IOL7	P0IOL6	P0IOL5	P0IOL4	P0IOL3	P0IOL2	P0IOL1	P0IOL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: F0H

P0IOL[7:6] P03 Function select

P0IOL7	P0IOL6	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode(Both input and output are disabled)

P0IOL[5:4] P02 Function select

P0IOL5	P0IOL4	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode(Both input and output are disabled)

P0IOL[3:2] P01 Function select

P0IOL3	P0IOL2	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode(Both input and output are disabled)

NOTE:

1. P01 is set as input mode when reset

P0IOL[1:0] P00 Function select

P0IOL1	P0IOL0	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode(Both input and output are disabled)

NOTE:

1. P00 is set as input mode when reset

P0DB (P0 Debounce Enable Register): B1H

7	6	5	4	3	2	1	0
DBCLK1	DBCLK0	–	–	P03DB	P02DB	P01DB	P00DB
R/W	R/W	–	–	R/W	R/W	R/W	R/W

Initial value: 00H

DBCLK[1:0] Configure Debounce Clock of Port

DBCLK1	DBCLK0	Description
0	0	fx/(SCLK)
0	1	fx/4
1	0	fx/16
1	1	fx/64

P03DB Configure Debounce of P03 Port

0	Disable
1	Enable

P02DB Configure Debounce of P02 Port

0	Disable
1	Enable

P01DB Configure Debounce of P01 Port

0	Disable
1	Enable

P00DB Configure Debounce of P00 Port

0	Disable
1	Enable

NOTES:

1. The signal is eliminated as noise if the same level is not detected on the enabled pin three or four times in a row at the sampling clock.
2. A pulse level should be input for three clocks or more to be detected as a valid edge.
3. The port debounce is automatically disabled at stop mode and recovered after stop mode release.

P0FSRH (Port 0 Function Selection High Register): B7H

7	6	5	4	3	2	1	0
–	P0FSRH6	–	P0FSRH4	P0FSRH3	P0FSRH2	–	P0FSRH0
–	R/W	–	R/W	R/W	R/W	–	R/W

Initial value: 00H

P0FSRH[6] P07 Function select
- 0 AN7 Function
- 1 LPDE Function

P0FSRH[4] P06 Function Select
- 0 AN6 Function
- 1 LPRXD0 Function

P0FSRL[3:2] P05 Function select

P0FSRL3	P0FSRL2	Description
0	0	AN5 Function
0	1	EC1 Function
1	0	LPTXD0 Function
1	1	Not available

P0FSRL[0] P00 Function select
- 0 AN4 Function
- 1 EC0 Function

P0FSRL (Port 0 Function Selection Low Register): B6H

7	6	5	4	3	2	1	0
–	P0FSRL6	–	P0FSRL4	P0FSRL3	P0FSRL2	P0FSRL1	P0FSRL0
–	R/W	–	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

P0FSRL[6] P03 Function select
- 0 AN3 Function
- 1 T1O/PWM1O Function

P0FSRL[4] P02 Function Select
- 0 AN2 Function
- 1 T0O/PWM0O Function

P0FSRL[3:2] P01 Function select

P0FSRL3	P0FSRL2	Description
0	0	AN1 Function
0	1	SCL0 Function
1	0	LPRXD0 Function
1	1	Not available

P0FSRL[1:0] P00 Function select

P0FSRL1	P0FSRL0	Description
0	0	AN0 Function
0	1	SDA0 Function
1	0	LPTXD0 Function
1	1	Not available

5.4 Port P1

5.4.1 Port Description of P1

As a 6-bit I/O port, P1 controls the following registers:

- P1 data register (P1)
- P1 direction registers (P1IOH/L)
- P1 pull-up resistor selection register (P1PU)
- P1 debounce enable register (P1DB)
- P1 open-drain selection register (P1OD)
- P1 Function selection registers (P1FSRH/P1FSRL)

5.4.2 Register Description of P1

P1 (P1 Data Register): 88H

7	6	5	4	3	2	1	0
–	–	P15	P14	P13	P12	P11	P10
–	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

P1[5:0]	I/O Data

P1PU (P1 Pull-up Resistor Selection Register): BDH

7	6	5	4	3	2	1	0
–	–	P15PU	P14PU	P13PU	P12PU	P11PU	P10PU
–	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

P1PU[5:0]	Configure Pull-up Resistor of P1 Port	
	0	Disable
	1	Enable

P1OD (P1 Open-drain Selection Register): BCH

7	6	5	4	3	2	1	0
–	–	P15OD	P14OD	P13OD	P12OD	P11OD	P10OD
–	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

P1OD[5:0]	Configure Open-drain of P1 Port	
	0	Push-pull output
	1	Open-drain output

P1IOH (P1 Direction High Register): BBH

7	6	5	4	3	2	1	0
–	–	–	–	P1IOH3	P1IOH2	P1IOH1	P1IOH0
–	–	–	–	R/W	R/W	R/W	R/W

Initial value: 0FH

P1IOH[3:2] P15 Function select

P1IOH3	P1IOH2	Description
0	0	Input mode (EINT3 function possible)
0	1	Output mode
1	0	Alternative function
1	1	Off mode(Both input and output are disabled)

P1IOH[1:0] P14 Function select

P1IOH1	P1IOH0	Description
0	0	Input mode (EINT2 function possible)
0	1	Output mode
1	0	Alternative function
1	1	Off mode (Both input and output are disabled)

P1IOL (P1 Direction Low Register): BAH

7	6	5	4	3	2	1	0
P1IOL7	P1IOL6	P1IOL5	P1IOL4	P1IOL3	P1IOL2	P1IOL1	P1IOL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 03H

P1IOL[7:6] P13 Function select

P1IOL7	P1IOL6	Description
0	0	Input mode (EINT12 function possible)
0	1	Output mode
1	0	Alternative function
1	1	Off mode (Both input and output are disabled)

NOTE:

1. P13 is set as input mode when reset

P1IOL[5:4] P12 Function select

P1IOL5	P1IOL4	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode (both input and output are disabled)

NOTE:

1. P12 is set as input mode when reset

P1IOL[3:2] P11 Function select

P1IOL3	P1IOL2	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode (both input and output are disabled)

NOTE:

1. P11 is set as input mode when reset

P1IOL[1:0] P10 Function select

P1IOL1	P1IOL0	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode (both input and output are disabled)

P1DB (P1 Debounce Enable Register): B9H

7	6	5	4	3	2	1	0
–	–	P15DB	P14DB	P13DB	–	–	–
–	–	R/W	R/W	R/W	–	–	–

Initial value: 00H

P15DB	Configure Debounce of P15 Port	
	0	Disable
	1	Enable
P14DB	Configure Debounce of P14 Port	
	0	Disable
	1	Enable
P13DB	Configure Debounce of P13 Port	
	0	Disable
	1	Enable

NOTES:

1. The signal is eliminated as noise if the same level is not detected on enabled pin three or four times in a row at the sampling clock.
2. A pulse level should be input for three clocks or more to be detected as a valid edge.
3. The port debounce is automatically disabled in stop mode and recovered after stop mode is released.
4. Refer to the port 0 debounce enable register (P0DB) for the debounce clock of port 1.

P1FSRH (Port 1 Function Selection High Register): BFH

7	6	5	4	3	2	1	0
–	–	–	–	–	P1FSRH2	–	P1FSRH0
–	–	–	–	–	R/W	–	R/W

Initial value: 00H

P1FSRH2	P15 Function select	
	0	MISO0 Function
	1	XIN Function
P1FSRH0	P14 Function select	
	0	MOSI0 Function
	1	XOUT Function

NOTES:

1. The pull-up resistor of P14/P15 is automatically disabled regardless of P14PU/P15PU value if the P14/P15 is configured as an X-tal function (XIN/XOUT).
2. The P1FSRH[2,0] and P1IOH[3:0] bits won't be changed during the fXIN is selected as the system clock (fx).

P1FSRL (Port 1 Function Selection Low Register): BEH

7	6	5	4	3	2	1	0
–	P1FSRL6	–	P1FSRL4	–	P1FSRL2	–	P1FSRL0
–	R/W	–	R/W	–	R/W	–	R/W

Initial value: 00H

P1FSRL6	P13 Function select	
	0	T2O/PWM2O Function
	1	RTCOUT Function
P1FSRL4	P12 Function select	
	0	SCL0 Function
	1	SXOUT Function
P1FSRL2	P11 Function select	
	0	SDA0 Function
	1	SXIN Function
P1FSRL0	P10 Function select	
	0	LPDE0 Function
	1	SCK0 Function

NOTES:

1. Refer to the configure option for the P10/RESETB.
2. The pull-up resistor of P11/P12 is automatically disabled regardless of P11PU/P12PU value if the P11/P12 is configured as an X-tal function (SXIN/SXOUT).
3. The P1FSRL[4,2] and P1IOL[5:2] bits won’t be changed during the fSUB is selected as system clock (fx).

5.5 Port P2

5.5.1 Port Description of P2

As a 4-bit I/O port, P2 controls the following registers:

- P2 data register (P2)
- P2 direction register (P2IOL)
- P2 pull-up resistor selection register (P2PU)
- P2 open-drain selection register (P2OD)
- P2 Function selection registers (P2FSRL)

5.5.2 Register Description of P2

P2 (P2 Data Register): 90H

7	6	5	4	3	2	1	0
–	–	–	–	P23	P22	P21	P20
–	–	–	–	R/W	R/W	R/W	R/W

Initial value: 00H

P2[3:0]	I/O Data

P2PU (P2 Pull-up Resistor Selection Register): C5H

7	6	5	4	3	2	1	0
–	–	–	–	P23PU	P22PU	P21PU	P20PU
–	–	–	–	R/W	R/W	R/W	R/W

Initial value: 00H

P2PU[3:0]	Configure Pull-up Resistor of P2 Port	
	0	Disable
	1	Enable

P2OD (P2 Open-drain Selection Register): C4H

7	6	5	4	3	2	1	0
–	–	–	–	P23OD	P22OD	P21OD	P20OD
–	–	–	–	R/W	R/W	R/W	R/W

Initial value: 00H

P2OD[3:0]	Configure Open-drain of P2 Port	
	0	Push-pull output
	1	Open-drain output

P2IOL (P2 Direction Low Register): C2H

7	6	5	4	3	2	1	0
P2IOL7	P2IOL6	P2IOL5	P2IOL4	P2IOL3	P2IOL2	P2IOL1	P2IOL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

P2IOL[7:6] P23 Function select

P2IOL7	P2IOL6	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode (both input and output are disabled)

P2IOL[5:4] P22 Function select

P2IOL5	P2IOL4	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode (both input and output are disabled)

P2IOL[3:2] P21 Function select

P2IOL3	P2IOL2	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode (both input and output are disabled)

P2IOL[1:0] P20 Function select

P2IOL1	P2IOL0	Description
0	0	Input mode
0	1	Output mode
1	0	Alternative function
1	1	Off mode (both input and output are disabled)

P2FSRL (Port 2 Function Selection Low Register): C6H

7	6	5	4	3	2	1	0
–	P2FSRL6	–	P2FSRL4	–	P2FSRL2	–	P2FSRL0
–	R/W	–	R/W	–	R/W	–	R/W

Initial value: 00H

P2FSRL6	P23 Function select	
	x	SCK0 Function
P2FSRL4	P22 Function select	
	0	SCL0 Function
	1	MISO0 Function
P2FSRL2	P21 Function select	
	0	SDA0 Function
	1	MOSI0 Function
P2FSRL0	P20 Function select	
	0	EC2 Function
	1	SS0 Function

6 Interrupt Controller

Up to 15 interrupt sources are available in the A96L116. Allowing software control, each interrupt source can be enabled by defining a separate enable register bit associated with it. It can also have four levels of priority assigned. The non-maskable interrupt source is always enabled with a higher priority than any other interrupt source and is not controllable by software.

The interrupt controller features the following:

- Receives requests from 15 interrupt sources.
- Priority of six groups
- Four priority levels
- Multi interrupt possibility
- If requests of different priority levels are received simultaneously, a request with a higher priority level is served first.
- Each interrupt source can be controlled by an EA bit and an IEx bit.
- Interrupt latency varies, ranging from 3 to 9 machine cycles in a single interrupt system.

A non-maskable interrupt is always enabled, while maskable interrupts can be enabled through four pairs of interrupts enable registers (IE, IE1, IE2, and IE3). Each bit of the four registers can individually enable or disable a particular interrupt source. Especially bit 7 (EA) in the register IE provides overall control. It must be set to ‘1’ to enable interrupts as described in the followings:

- When EA is set to ‘0’ ➔ All interrupts are disabled.
- When EA is set to ‘1’ ➔ A particular interrupt can be individually enabled or disabled by the associate bit of the interrupt enable registers.

EA is always cleared to ‘0’, jumping to an interrupt service vector and set to ‘1’ executing the [RETI] instruction. A96L116 supports a four-level priority scheme. Each maskable interrupt is individually assigned to one of the four levels according to IP and IP1.

Figure 14. Interrupt Group Priority Level

Interrupt Group	Highest → Lowest			
0 (Bit0)	Interrupt 0	Interrupt 6	Interrupt 12	Interrupt 18
1 (Bit1)	Interrupt 1	Interrupt 7	Interrupt 13	Interrupt 19
2 (Bit2)	Interrupt 2	Interrupt 8	Interrupt 14	Interrupt 20
3 (Bit3)	Interrupt 3	Interrupt 9	Interrupt 15	Interrupt 21
4 (Bit4)	Interrupt 4	Interrupt 10	Interrupt 16	Interrupt 22
5 (Bit5)	Interrupt 5	Interrupt 11	Interrupt 17	Interrupt 23

Highest

Lowest

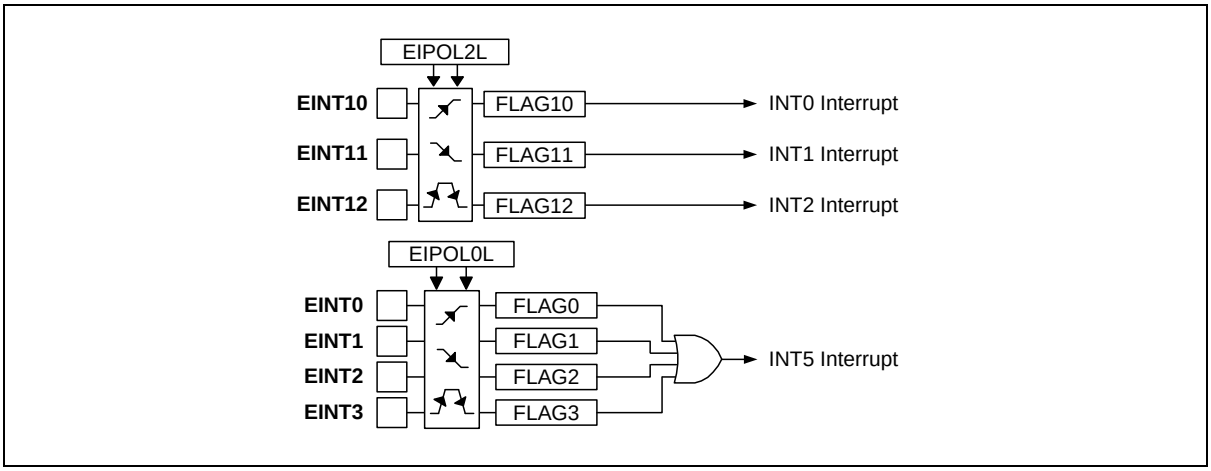
Figure 14 describes interrupt groups and their priority levels that are available for sharing interrupt priority. The priority of a group is set by two bits of Interrupt Priority (IP) registers: one bit from IP and another one bit from IP1.

Interrupt Service Routine serves an interrupt having higher priority first. If two requests of different priority levels are received simultaneously, the request with a higher priority level is served before the lower one.

6.1 External Interrupt

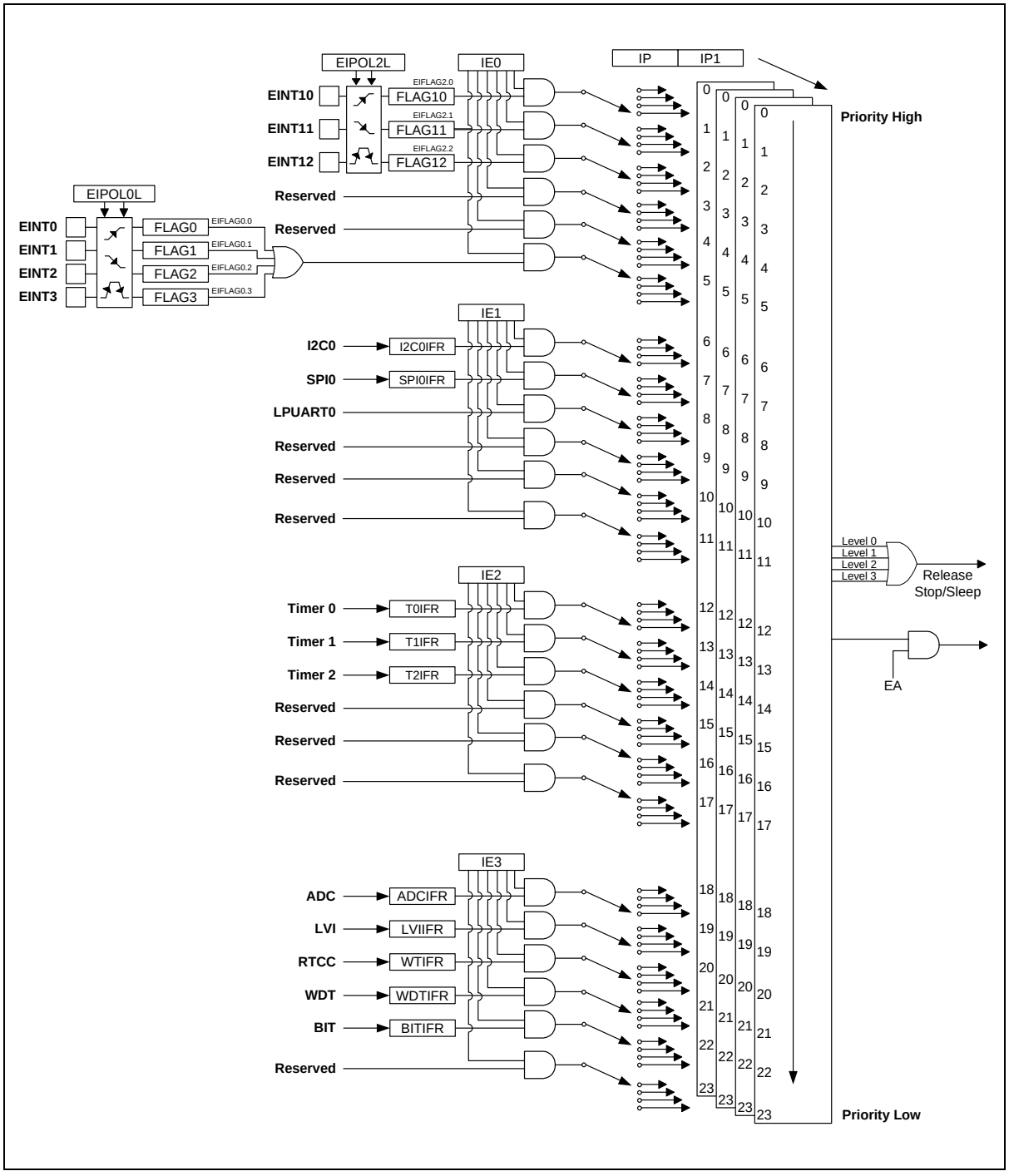
External interrupts on pins of INT0 to INT5 receive various interrupt requests depending on the external interrupt polarity 0 Low register (EIPOL0L) and external interrupt polarity 2 Low register (EIPOL2L), as shown in Figure 15. Each external interrupt source has enable/disable bits. An external interrupt flag register (EIFLAG) provides the status of the external interrupts.

Figure 15. External Interrupt Description



6.2 Interrupt Controller Block Diagram

Figure 16. Interrupt Controller Block Diagram



In Figure 16, the release signal for STOP and IDLE modes can be generated by all interrupt sources enabled without reference to the priority level. An interrupt request will be delayed while data is written to one of the registers IE, IE1, IE2, IE3, IP, IP1, and PCON.

6.3 Interrupt Vector Table

When a certain interrupt occurs, an LCALL (Long Call) instruction pushes the contents of the PC (Program Counter) onto the stack and loads the appropriate vector address. CPU pauses from its current task for some time and processes the interrupt at the vector address.

The interrupt controller supports 24 interrupt sources, each with a determined priority order, as shown in Table 9.

Table 9. Interrupt Vector Address Table

Interrupt Source	Symbol	Interrupt Enable Bit	Priority	Mask	Vector Address
Hardware RESET	RESETB	-	0	Non-Maskable	0000H
External Interrupt 10	INT0	IE.0	1	Maskable	0003H
External Interrupt 11	INT1	IE.1	2	Maskable	000BH
External Interrupt 12	INT2	IE.2	3	Maskable	0013H
–	INT3	IE.3	4	Maskable	001BH
–	INT4	IE.4	5	Maskable	0023H
External Interrupt 0 – 3	INT5	IE.5	6	Maskable	002BH
I2C0 Interrupt	INT6	IE1.0	7	Maskable	0033H
SPI0 Interrupt	INT7	IE1.1	8	Maskable	003BH
LPUART0 Interrupt	INT8	IE1.2	9	Maskable	0043H
–	INT9	IE1.3	10	Maskable	004BH
–	INT10	IE1.4	11	Maskable	0053H
–	INT11	IE1.5	12	Maskable	005BH
T0 Match Interrupt	INT12	IE2.0	13	Maskable	0063H
T1 Match Interrupt	INT13	IE2.1	14	Maskable	006BH
T2 Match Interrupt	INT14	IE2.2	15	Maskable	0073H
–	INT15	IE2.3	16	Maskable	007BH
–	INT16	IE2.4	17	Maskable	0083H
–	INT17	IE2.5	18	Maskable	008BH
ADC Interrupt	INT18	IE3.0	19	Maskable	0093H
LVI Interrupt	INT19	IE3.1	20	Maskable	009BH
RTCC Interrupt	INT20	IE3.2	21	Maskable	00A3H
WDT Interrupt	INT21	IE3.3	22	Maskable	00ABH
BIT Interrupt	INT22	IE3.4	23	Maskable	00B3H
–	INT23	IE3.5	24	Maskable	00BBH

To execute the maskable interrupts, both the EA bit and a corresponding bit of Iex associated with a specific interrupt source must be set to '1'. When an interrupt request is received, a particular interrupt request flag is set to '1' and maintains its status until the CPU accepts the interrupt. After the interrupt acceptance, the interrupt request flag will be cleared automatically.

6.4 Interrupt Sequence

An interrupt request is held until the interrupt is accepted or the interrupt latch is cleared to '0' by a reset or an instruction. The interrupt acceptance always happens at the last cycle of the instruction process. So, rather than fetching the current instruction, the CPU executes internal LCALL instruction and saves a PC onto the stack.

To begin an ISR (Interrupt Service Routine), the interrupt controller uses a branch instruction LJMP (Long Jump). The interrupt controller gives the address of LJMP instruction to the CPU. Since the end of the execution of the current instruction, it needs 3~9 machine cycles to go to the interrupt service routine. The interrupt service task is terminated by the interrupt return instruction [RETI]. Once an interrupt request is generated, the following process is performed.

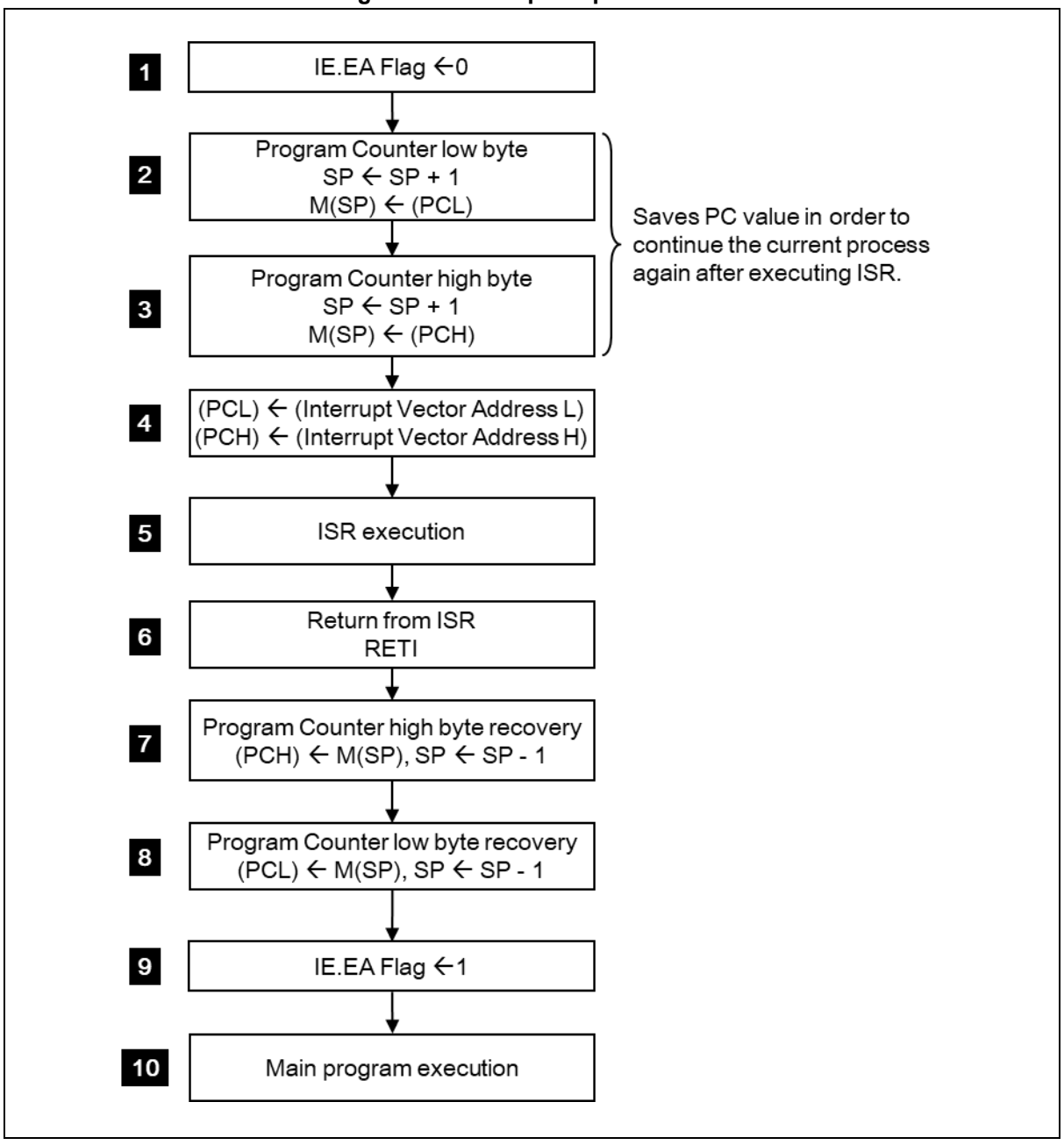
Table 10 describes the LJMP example code.

Table 10. LJMP Description and Example Code

<table>
<tr><th>Instruction</th><td colspan="3">LJMP</td></tr>
<tr><th rowspan="2">Example code</th><td colspan="3">LJMP 4000H</td></tr>
<tr><td colspan="3">
<table>
<tr><th>Address</th><th>Data</th><th>Instruction</th></tr>
<tr><td>1280H</td><td>02</td><td rowspan="3">LJMP 4000H</td></tr>
<tr><td>1281H</td><td>40</td></tr>
<tr><td>1282H</td><td>00</td></tr>
<tr><td>1283H</td><td>E4</td><td>CLR A</td></tr>
<tr><td>⋮</td><td>⋮</td><td>⋮</td></tr>
<tr><td>4000H</td><td>00</td><td>NOP</td></tr>
<tr><td>4001H</td><td>23</td><td>RL A</td></tr>
<tr><td></td><td></td><td></td></tr>
</table>
NOTE:
1. After finishing LJMP, NOP located at the address 400H will be executed as the next instruction.
</td></tr>
</table>

Figure 17 shows a flow diagram of an ISR process.

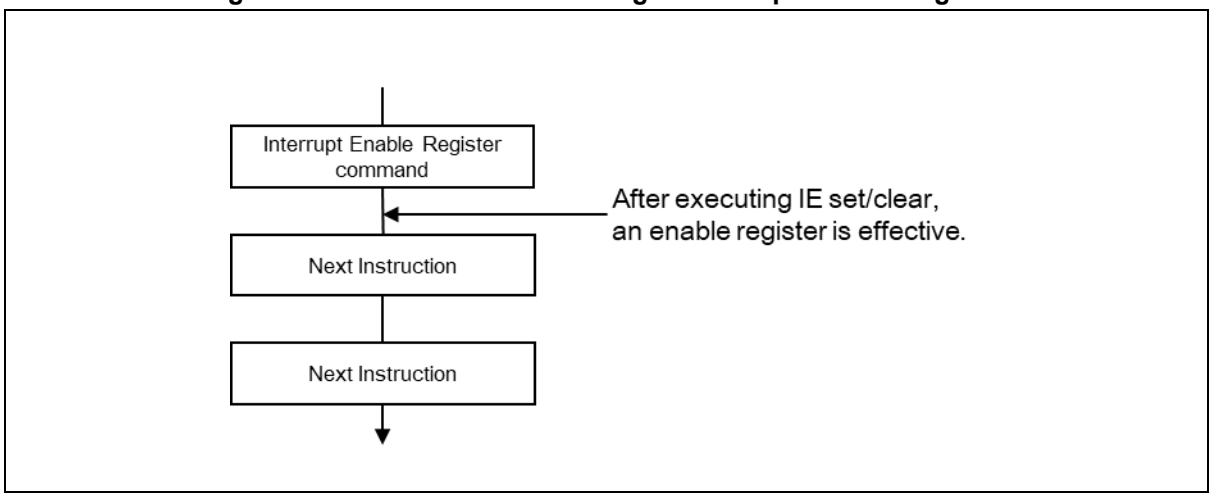
Figure 17. Interrupt Sequence Flow



6.5 Effective Timing after Controlling Interrupt Bit

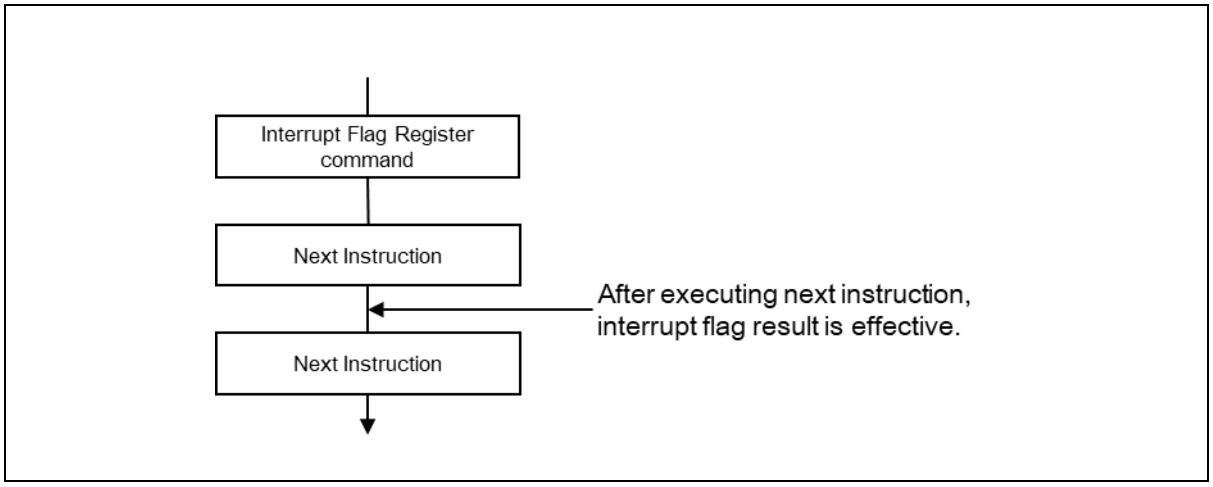
Case A in Figure 18 shows the effective time of the Control Interrupt Enable Register (IE, IE1, IE2, and IE3).

Figure 18. Case A: Effective Timing of Interrupt Enable Register



Case B in Figure 19 shows the effective time of the Interrupt Flag Register.

Figure 19. Case B: Effective Timing of Interrupt Flag Register



6.6 Multi-Interrupt

If two requests of different priority levels are received simultaneously, the request with a higher priority level is served first. If more than one interrupt request is received, the interrupt polling sequence determines which request is served first by hardware. However, for special features, multi-interrupt processing can be executed by software.

Figure 20. Effective Timing of Multi-Interrupt

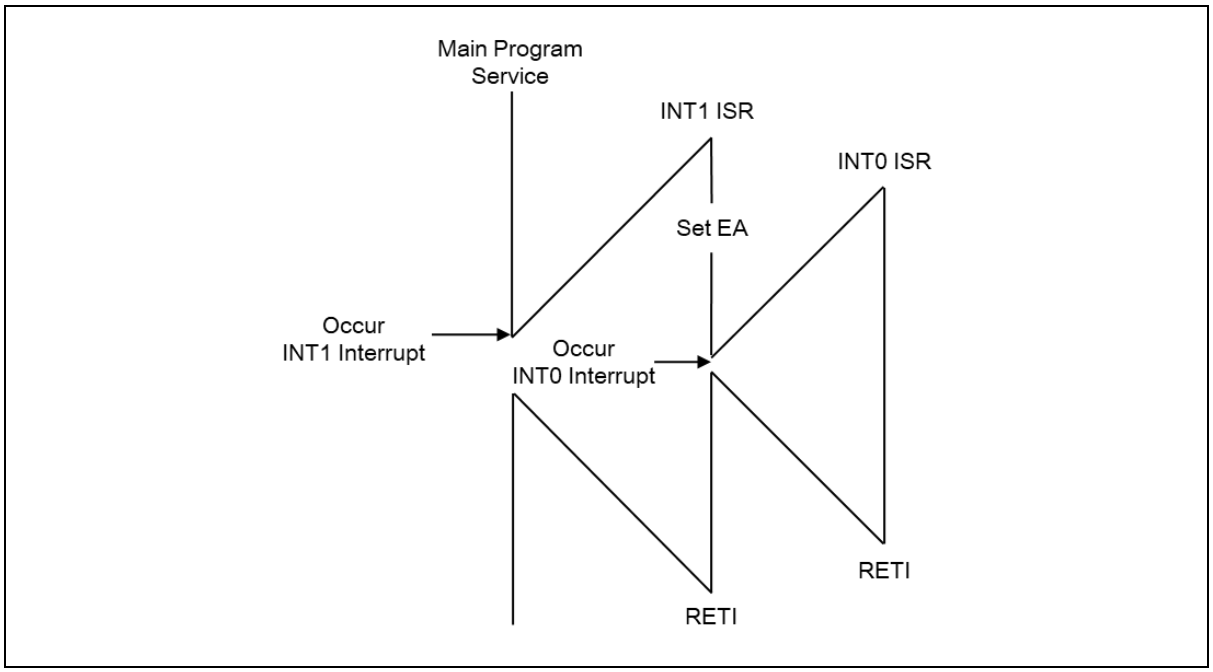


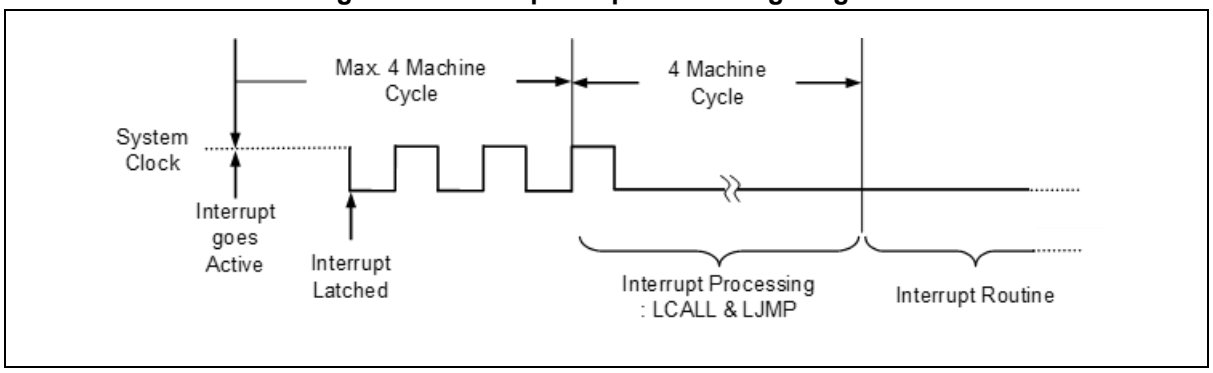
Figure 20 shows an example of multi-interrupt processing. While INT1 is served, INT0, which has higher priority than INT1, occurs. Then, INT0 is served immediately, and the remaining part of the INT1 service routine is executed. If the priority level of INT0 is the same or lower than INT1, INT0 will be served after the INT1 service has been completed.

An interrupt service routine can be interrupted only by an interrupt with higher priority. If two interrupts of different priority occur simultaneously, the interrupt with a higher priority level will be served first. An interrupt cannot be interrupted by another interrupt with the same or a lower priority level. If two interrupts with the same priority level occur simultaneously, the scan order will determine the service order for those interrupts.

6.7 Interrupt Enable Accept Timing

Figure 21 implies that some time is required to respond to the latched interrupt signal. In this figure, four machine cycles will be taken for the processes of LCALL and LJMP.

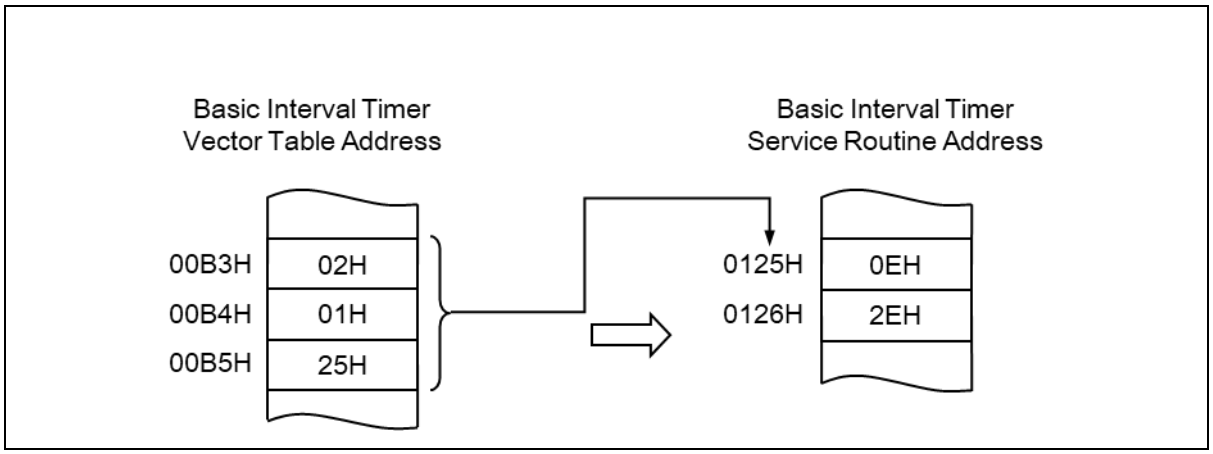
Figure 21. Interrupt Response Timing Diagram



6.8 Interrupt Service Routine Address

As shown in Figure 22, ISR can be placed at any other location in program memory, and program memory must provide an unconditional jump to the starting address of ISR from the corresponding vector address.

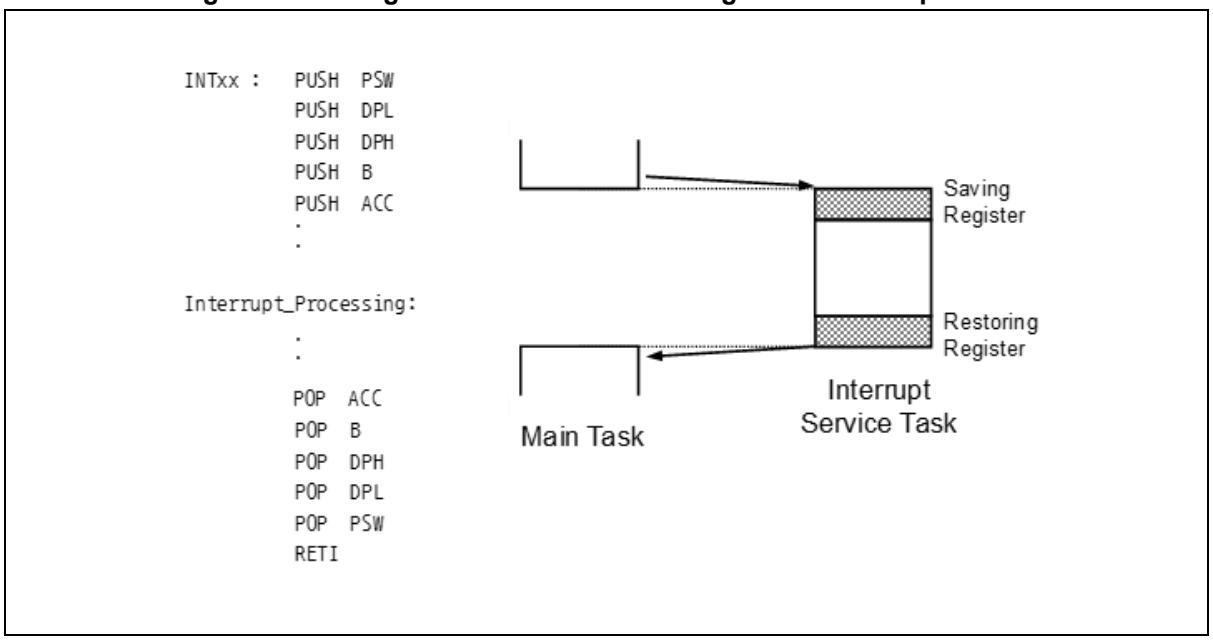
Figure 22. Correspondence between Vector Table Address and ISR Entry Address



6.9 Saving/Restore General-Purpose Registers

Assumimg an emergency occurs, CPU must pause the current task (main task in Figure 23) to execute another task (interrupt service Task in Figure 23). After finishing another task, the CPU returns to the current task (main task).

Figure 23. Saving and Restore Process Diagram and Example Code



The example code in Figure 23 performs the followings:

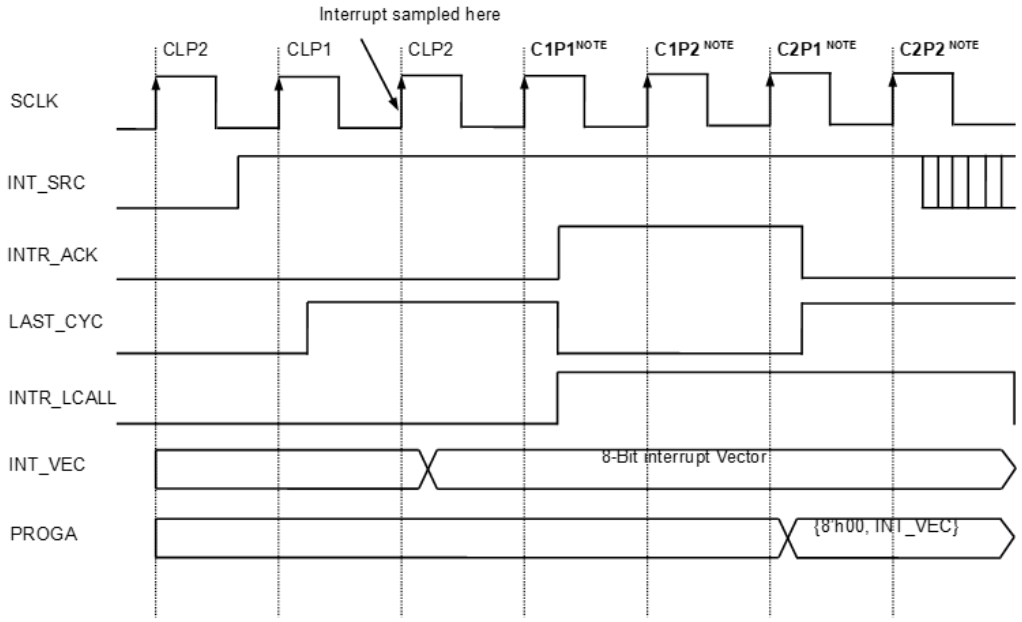
1. Interrupt INTxx occurs.
2. PUSH PSW: the SP is incremented by one, and the value of the specified byte operand is stored at the internal RAM address indirectly referenced by the SP.
3. PUSH DPL, PUSH DPH: PSW in memory stack with the help of PUSH instruction.
4. B and ACC in memory stack with the help of a PUSH instruction
5. CPU pops the value of the flag register and stores it in register H with the help of POP Instruction.

6.10 Interrupt Timing

As seen in Figure 24 below, an interrupt source is sampled at the last cycle of a command. Upon the sampling, a low 8-bit interrupt vector is decided.

M8051W core makes the interrupt acknowledge at the first command cycle and executes LCALL instruction to jump interrupt routine at the address referenced by INT_VEC.

Figure 24. Timing Chart of Interrupt Acceptance and Interrupt Return Instruction



NOTE:

1. Command cycle CLPx implies the following:
   - L ➔ Last cycle
   - 1 ➔ 1st cycle or 1st phase
   - 2 ➔ 2nd cycle or 2nd phase

6.11 Interrupt Register

Interrupt registers are memory spaces used to control interrupt functions. As shown in Table 11, the interrupt registers consist of Interrupt Enable Registers, Interrupt Priority Registers, External Interrupt Flag Registers, and External Interrupt Polarity Register.

6.11.1 Interrupt Enable Registers (IE, IE1, IE2, IE3)

Interrupt enable register comprises a global interrupt control bit (EA) and peripheral interrupt control bits. A total of 24 peripherals can control interrupts.

6.11.2 Interrupt Priority Registers (IP, IP1)

Twenty-four interrupts are divided into six groups, each with four interrupt sources. A group can be assigned four levels of interrupt priority by using an interrupt priority register. Level 3 is the highest priority, while level 0 is the lowest. After a reset, IP and IP1 are cleared to ‘00H’. If interrupts have the same priority level, a lower numbered interrupt is served first.

6.11.3 External Interrupt Flag Register (EIFLAG0, EIFLAG1, EIFLAG2)

External interrupt flag (EIFLAG) is set to ‘1’ when the external interrupt generating condition is satisfied. The flag is cleared when the interrupt service routine is executed. Alternatively, the flag can be cleared by writing ‘0’.

6.11.4 External Interrupt Polarity Registers (EIPOL0L, EIPOL2L)

External interrupt polarity 0 Low register (EIPOL0L) and external interrupt polarity 2 Low register (EIPOL2L) determine one from rising edge, falling edge, and both edges for interrupt. No interrupt is at any edge by default.

6.11.5 Register Map

Table 11. Interrupt Register Map

Name	Address	Direction	Default	Description
IE	A8H	R/W	00H	Interrupt Enable Register
IE1	A9H	R/W	00H	Interrupt Enable Register 1
IE2	AAH	R/W	00H	Interrupt Enable Register 2
IE3	ABH	R/W	00H	Interrupt Enable Register 3
IP	B8H	R/W	00H	Interrupt Priority Register
IP1	F8H	R/W	00H	Interrupt Priority Register 1
EIFLAG0	91H	R/W	00H	External Interrupt Flag 0 Register
EIFLAG1	99H	R/W	00H	External Interrupt Flag 1 Register
EIFLAG2	A1H	R/W	00H	External Interrupt Flag 2 Register
EIPOL0L	9AH	R/W	00H	External Interrupt Polarity 0 Low Register
EIPOL2L	ACH	R/W	00H	External Interrupt Polarity 2 Low Register

6.11.6 Interrupt Register Description

IE (Interrupt Enable Register): A8H

7	6	5	4	3	2	1	0
EA	–	INT5E	–	–	INT2E	INT1E	INT0E
R/W	–	R/W	–	–	R/W	R/W	R/W

Initial value: 00H

EA Enable or Disable All Interrupt bits
- 0 All Interrupt disable
- 1 All Interrupt enable

INT5E Enable or Disable External Interrupt 0 – 3 (EINT0 – EINT3)
- 0 Disable
- 1 Enable

INT2E Enable or Disable External Interrupt 12 (EINT12)
- 0 Disable
- 1 Enable

INT1E Enable or Disable External Interrupt 11 (EINT11)
- 0 Disable
- 1 Enable

INT0E Enable or Disable External Interrupt 10 (EINT10)
- 0 Disable
- 1 Enable

IE1 (Interrupt Enable Register 1): A9H

7	6	5	4	3	2	1	0
–-	–	–	–	–	INT8E	INT7E	INT6E
–	–	–	–	–	R/W	R/W	R/W

Initial value: 00H

INT8E Enable or Disable LPUART0 Interrupt
- 0 Disable
- 1 Enable

INT7E Enable or Disable SPI0 Interrupt
- 0 Disable
- 1 Enable

INT6E Enable or Disable I2C0 Interrupt
- 0 Disable
- 1 Enable

IE2 (Interrupt Enable Register 2): AAH

7	6	5	4	3	2	1	0
–	–	–	–	–	INT14E	INT13E	INT12E
–	–	–	–	–	R/W	R/W	R/W

Initial value: 00H

INT14E	Enable or Disable Timer 2 Interrupt	
	0	Disable
	1	Enable
INT13E	Enable or Disable Timer 1 Interrupt	
	0	Disable
	1	Enable
INT12E	Enable or Disable Timer 0 Interrupt	
	0	Disable
	1	Enable

IE3 (Interrupt Enable Register 3): ABH

7	6	5	4	3	2	1	0
–	–	–	INT22E	INT21E	INT20E	INT19E	INT18E
–	–	–	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

INT22E	Enable or Disable BIT Interrupt	
	0	Disable
	1	Enable
INT21E	Enable or Disable WDT Interrupt	
	0	Disable
	1	Enable
INT20E	Enable or Disable RTCC Interrupt	
	0	Disable
	1	Enable
INT19E	Enable or Disable LVI Interrupt	
	0	Disable
	1	Enable
INT18E	Enable or Disable ADC Interrupt	
	0	Disable
	1	Enable

IP (Interrupt Priority Register): B8H

7	6	5	4	3	2	1	0
–	–	IP5	IP4	IP3	IP2	IP1	IP0
–	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

IP1 (Interrupt Priority Register 1): F8H

7	6	5	4	3	2	1	0
–	–	IP15	IP14	IP13	IP12	IP11	IP10
–	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

IP[5:0], IP1[5:0] Select Interrupt Group Priority

IP1x	Ipx	Description
0	0	level 0 (lowest)
0	1	level 1
1	0	level 2
1	1	level 3 (highest)

EIFLAG0 (External Interrupt Flag 0 Register): 91H

7	6	5	4	3	2	1	0
–	–	–	–	FLAG3	FLAG2	FLAG1	FLAG0
–	–	–	–	R/W	R/W	R/W	R/W

Initial value: 00H

EIFLAG0[3:0] When an external interrupt occurs, the flag becomes ‘1’. The flag is cleared only by writing a ‘0’ to the bit. So, the flag should be cleared by software.

0 External Interrupt did not occur.

1 External Interrupt occurred.

EIPOL0L (External Interrupt Polarity 0 Low Register): 9AH

7	6	5	4	3	2	1	0
POL3		POL2		POL1		POL0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

EIPOL0L[3:0] External interrupt (EINT0 ,,,,,,, EINT3) polarity selection

POLn[3:0]		Description
0	0	No interrupt at any edge
0	1	Interrupt on the rising edge
1	0	Interrupt on the falling edge
1	1	Interrupt on both of rising and falling edge

Where n=0, 1, 2, and 3

EIFLAG1 (External Interrupt Flag 1 Register): 99H

7	6	5	4	3	2	1	0
I2C0IFR	–	–	–	–	–	–	–
R/W	–	–	–	–	–	–	–

Initial value: 00H

I2C0IFSR This bit is an I2C0 interrupt flag bit. When the interrupt occurs, this bit becomes '1'. This bit is cleared when writing any values in the I2C0SR register.

0 I2C0 Interrupt did not occur.

1 I2C0 Interrupt occurred.

EIFLAG2 (External Interrupt Flag 2 Register): A1H

7	6	5	4	3	2	1	0
–	–	–	–	–	FLAG12	FLAG11	FLAG10
–	–	–	–	–	R/W	R/W	R/W

Initial value: 00H

FLAG12 When an external interrupt 12 occurs, the flag becomes '1'.
The flag is cleared by writing '0' to the bit or automatically cleared by the INT_ACK signal.

0 External Interrupt did not occur.

1 External Interrupt occurred

FLAG11 When an external interrupt 11 occurs, the flag becomes '1'.
The flag is cleared by writing '0' to the bit or automatically cleared by the INT_ACK signal.

0 External Interrupt did not occur.

1 External Interrupt occurred.

FLAG10 When an external interrupt 10 occurs, the flag becomes '1'.
The flag is cleared by writing '0' to the bit or automatically cleared by the INT_ACK signal.

0 External Interrupt did not occur.

1 External Interrupt occurred.

EIPOL2L (External Interrupt Polarity 2 Low Register): ACH

7	6	5	4	3	2	1	0
–-	–	POL12		POL11		POL10	
–	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

EIPOL2L[5:0] External interrupt (EINT10 to EINT12) polarity selection

POLn[1:0]		Description
0	0	No interrupt at any edge
0	1	Interrupt on the rising edge
1	0	Interrupt on the falling edge
1	1	Interrupt on both of rising and falling edge

Where n=10, 11, and 12

7 Clock Generator

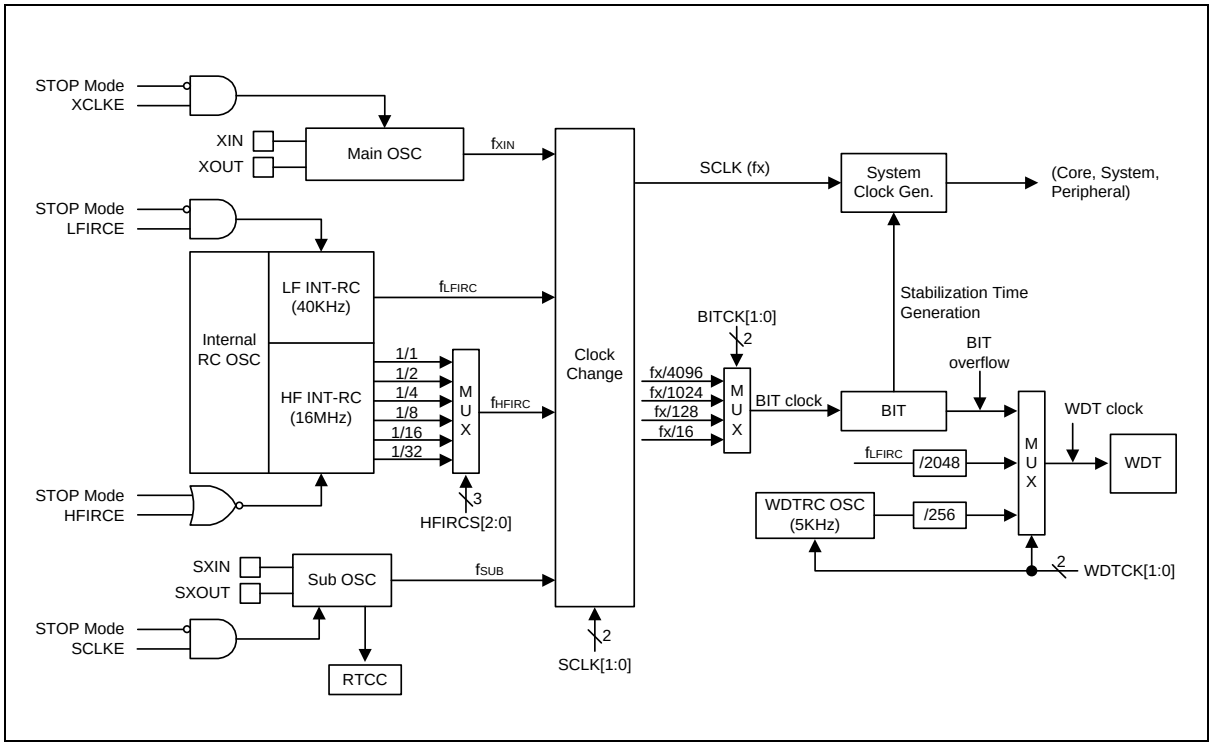
As shown in Figure 25, a clock generator produces basic clock pulses, providing a CPU and peripherals with a system clock. A default system clock is a 1 MHz HF INT-RC oscillator with sixteen default division rates. To stabilize the system internally, a 1 MHz HF INT-RC oscillator is used on POR.

A96L116 incorporates three types of oscillators:

- Calibrated HF Internal RC Oscillator (16 MHz)
  - HF INT-RC OSC/32 (0.5 MHz)
  - HF INT-RC OSC/16 (1 MHz, default system clock)
  - HF INT-RC OSC/8 (2 MHz)
  - HF INT-RC OSC/4 (4 MHz)
  - HF INT-RC OSC/2 (8 MHz)
  - HF INT-RC OSC/1 (16 MHz)
- Internal WDTRC Oscillator (5 kHz)
- LF INT-RC Oscillator (40 kHz)

7.1 Block Diagram

Figure 25. Clock Generator in Block Diagram



7.2 Register Map

Table 12. Clock Generator Register Map

Name	Address	Direction	Default	Description
SCCR	8AH	R/W	00H	System and Clock Control Register
OSCCR	C8H	R/W	08H	Oscillator Control Register
IRCTCR	8FH	R/W	00H	Internal RC Trim Control Register
IRCCTRM	1000H (XSFR)	R/W	0xH	Internal RC Coarse Trim Register
IRCFTRM	1001H (XSFR)	R/W	xxH	Internal RC Fine Trim Register
IRCIDR	9FH	R/W	00H	Internal RC Identification Register
SUBISET	C9H	R/W	07H	Sub Oscillator Driving Current Selection Register
PPCLKEN0	D1H	R/W	00H	Peripheral Clock Enable Register 0
PPCLKEN1	D9H	R/W	00H	Peripheral Clock Enable Register 1
PPCLKEN2	E1H	R/W	00H	Peripheral Clock Enable Register 2
PPCLKEN3	E9H	R/W	00H	Peripheral Clock Enable Register 3
MPWRCR	F1H	R/W	00H	Memory Power Control Register

7.3 Register Description

SCCR (System and Clock Control Register): 8AH

7	6	5	4	3	2	1	0
–	–	–	–	–	–	SCLK1	SCLK0
–	–	–	–	–	–	R/W	R/W

Initial value: 00H

SCLK[1:0] System Clock Selection Bit

SCLK1	SCLK0	Description
0	0	HF INT-IRC OSC (f_{HFIRC}) for system clock
0	1	External Main OSC (f_{XIN}) for system clock
1	0	External Sub OSC (f_{SXIN}) for system clock
1	1	LF INT-IRC OSC (f_{LFIRC}) for system clock

NOTE:

1. If the OSCCR register disables a target system clock, the SCCR register won't be changed in case of selecting the corresponding clock as a system clock.

OSCCR (Oscillator Control Register): C8H

7	6	5	4	3	2	1	0
LFIRCE	–	HFIRCS2	HFIRCS1	HFIRCS0	HFIRCE	XCLKE	SCLKE
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 08H

LFIRCE — It controls the operation of the low frequency internal RC oscillator.
- 0 Disable operation of LFIRC OSC
- 1 Enable operation of LFIRC OSC

HFIRCS[2:0] — HFIRC oscillator post-divider selection

HFIRCS2	HFIRCS1	HFIRCS0	Description
0	0	0	f_{HFIRC}/32 (0.5 MHz)
0	0	1	f_{HFIRC}/16 (1 MHz)
0	1	0	f_{HFIRC}/8 (2 MHz)
0	1	1	f_{HFIRC}/4 (4 MHz)
1	0	0	f_{HFIRC}/2 (8 MHz)
1	0	1	f_{HFIRC}/1 (16 MHz)
Other values			Not used

HFIRCE — It controls the operation of the high frequency internal RC oscillator.
- 0 Enable operation of HFIRC OSC
- 1 Disable operation of HFIRC OSC

XCLKE — It controls the operation of the external main oscillator.
- 0 Disable operation of X-TAL
- 1 Enable operation of X-TAL

SCLKE — It controls the operation of the external sub-oscillator.
- 0 Disable operation of SX-TAL
- 1 Enable operation of SX-TAL

NOTE:

1. The system clock is not disabled by the corresponding bit of the OSCCR register.
Ex) The high frequency internal RC oscillator won't be disabled by the HFIRCE bit during the f_{HFIRC} is selected as the system clock.

IRCIDR (Internal RC Trim Identification Register): 9FH

7	6	5	4	3	2	1	0
IRCID7	IRCID6	IRCID5	IRCID4	IRCID3	IRCID2	IRCID1	IRCID0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

IRCID[7:0] — Internal RC Trim Identification.
- Others No identification value
- 01000110b Identification value for IRC Trim

(These bits are automatically cleared to logic '00H' immediately after one-time operation.)

IRCCTRM (Internal RC Course Trim Register): 1000H (XSFR)

7	6	5	4	3	2	1	0
–	–	–	–	–	CTRMH2	CTRMH1	CTRMH0
–	–	–	–	–	R/W	R/W	R/W

Initial value: 0xH

CTRMH[2:0] HFIRC course Trim bits.

These bits are read from “Configure Area” when a system reset occurs. These bits provide a user programmable trimming value on operation. The range is -4 to +3, the CTRMH2 is sign bit, and the frequency is changed by 1.3 [MHz] step-by-step. This register can be written with valid ID value and IRCTCR=0xB3.

IRCFTRM (Internal RC Fine Trim Register): 1001H (XSFR)

7	6	5	4	3	2	1	0
–	–	FTRMH5	FTRMH4	FTRMH3	FTRMH2	FTRMH1	FTRMH0
–	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: xxH

FTRMH[5:0] HFIRC Fine Trim bits.

These bits are read from “Configure Area” when a system reset occurs. These bits provide a user programmable trimming value on operation. The range is -32 to +31, the FTRMH5 is sign bit, and the frequency is changed by 80 [kHz] step-by-step. This register can be written with valid ID value and IRCTCR=0xB3.

IRCTCR (Internal RC Trim Control Register): 8FH

7	6	5	4	3	2	1	0
ITCR7	ITCR6	ITCR5	ITCR4	ITCR3	ITCR2	ITCR1	ITCR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

ITCR[7:0] Internal RC Trim Control Register.

Others	No effect
10110011b	IRCTRM register is used for IRC frequency. This register can be written with valid ID value.

SUBISET (Sub Oscillator Driving Current Selection Register): C9H

7	6	5	4	3	2	1	0
–	–	–	–	–	ISET2	ISET1	ISET0
–	–	–	–	–	R/W	R/W	R/W

Initial value: 07H

ISET[2:0] Sub Oscillator Driving Current Selection bits

ISET2	ISET1	ISET0	Description
0	0	0	Reserved (3rd level driving current)
0	0	1	Reserved (3rd level driving current)
0	1	0	3rd level driving current
0	1	1	4th level driving current
1	0	0	5th level driving current
1	0	1	6th level driving current
1	1	0	7th level driving current
1	1	1	The highest driving current

NOTE:

1. The “111b” should be set when the sub oscillator is started by s/w and the value should be kept during sub oscillator stabilization.

PPCLKEN0 (Peripheral Clock Control Register 0): D1H

7	6	5	4	3	2	1	0
RTCLKE	ADCLKE	–	–	–	P2CLK2	P1CLKE	P0CLKE
R/W	R/W	–	–	–	R/W	R/W	R/W

Initial value: 00H

RTCLKE RTCC Clock Enable bit
- 0 Disable RTCC clock
- 1 Enable RTCC clock

ADCLKE A/D Converter Clock Enable bit
- 0 Disable A/D converter clock
- 1 Enable A/D converter clock

P2CLKE Port 2 Clock Enable bit
- 0 Disable P2 clock
- 1 Enable P2 clock

P1CLKE Port 1 Clock Enable bit
- 0 Disable P1 clock
- 1 Enable P1 clock

P0CLKE Port 0 Clock Enable bit
- 0 Disable P0 clock
- 1 Enable P0 clock

NOTE:

1. The peripheral registers may not be read/written by software when the peripheral clock is disabled.

PPCLKEN1 (Peripheral Clock Control Register 1): D9H

7	6	5	4	3	2	1	0
CRCLKE	LVICLKE	–	–	–	T2CLK2	T1CLKE	T0CLKE
R/W	R/W	–	–	–	R/W	R/W	R/W

Initial value: 00H

CRCLKE	Flash CRC Clock Enable bit	
	0	Disable RTCC clock
	1	Enable RTCC clock
LVICLKE	Low-Voltage Indicator Clock Enable bit	
	0	Disable Low-Voltage Indicator clock
	1	Enable Low-Voltage Indicator clock
T2CLKE	Timer 2 Clock Enable bit	
	0	Disable timer 2 clock
	1	Enable timer 2 clock
T1CLKE	Timer 1 Clock Enable bit	
	0	Disable timer 1 clock
	1	Enable timer 1 clock
T0CLKE	Timer 0 Clock Enable bit	
	0	Disable timer 0 clock
	1	Enable timer 0 clock

NOTE:

1. The peripheral registers may not be read/written by software when the peripheral clock is disabled.

PPCLKEN2 (Peripheral Clock Control Register 2): E1H

7	6	5	4	3	2	1	0
LPUT0CLKE	–	–	–	SPI0CLKE	–	–	I2C0CLKE
R/W	–	–	–	R/W	–	–	R/W

Initial value: 00H

LPUT0CLKE	Low Power UART0 Clock Enable bit	
	0	Disable LPUART0 clock
	1	Enable LPUART0 clock
SPI0CLKE	SPI0 Clock Enable bit	
	0	Disable SPI0 clock
	1	Enable SPI0 clock
I2C0CLKE	I2C0 Clock Enable bit	
	0	Disable I2C0 clock
	1	Enable I2C0 clock

NOTE:

1. The peripheral registers may not be read/written by software when the peripheral clock is disabled.

PPCLKEN3 (Peripheral Clock Control Register 3): E9H

7	6	5	4	3	2	1	0
DFMCLKE	FMCLKE	–	LVRCLKE	–	–	–	WDTCLKE
R/W	R/W	–	R/W	–	–	–	R/W

Initial value: 00H

DFMCLKE	Data Flash Memory Control Clock Enable bit	
	0	Disable Data Flash memory control clock
	1	Enable Data Flash memory control clock
FMCLKE	Flash Memory Control Clock Enable bit	
	0	Disable flash memory control clock
	1	Enable flash memory control clock
LVRCLKE	Low-Voltage Reset Clock Enable bit	
	0	Disable LVR clock
	1	Enable LVR clock
WDTCLKE	Watchdog Timer Clock Enable bit	
	0	Disable WDT clock
	1	Enable WDT clock

NOTE:

1. The peripheral registers may not be read/written by software when the peripheral clock is disabled.

MPWRCR (Memory Power Control Register): F1H

7	6	5	4	3	2	1	0
–	–	–	DFMPWR	–	–	–	–
–	–	–	R/W	–	–	–	–

Initial value: 00H

DFMPWR	Data Flash Memory Power Control bit	
	0	Power-on
	1	Power-off (data flash memory has 2 µseconds power-on time. So, the software should wait at least 2 µseconds before data flash memory access after power-on)

8 Basic Interval Timer (BIT)

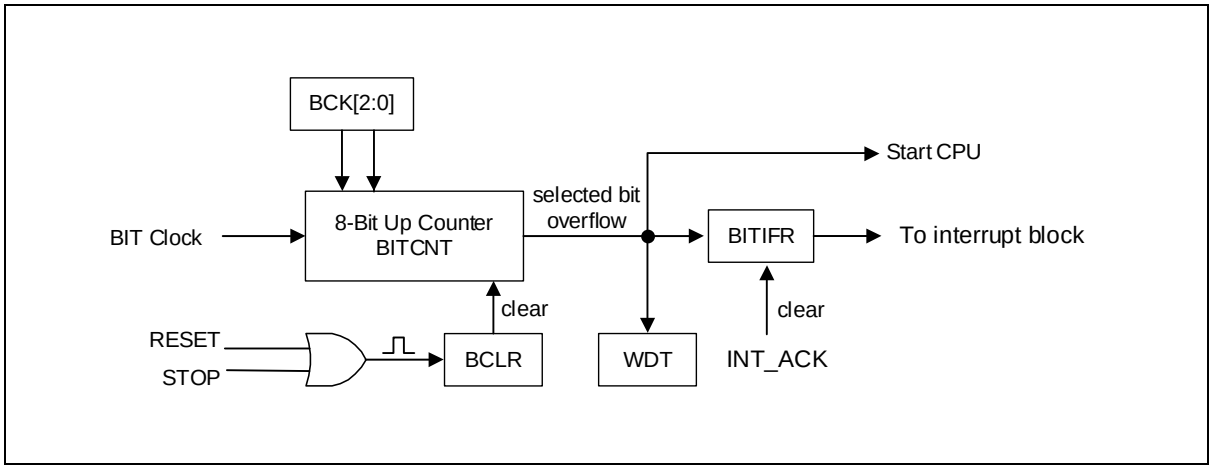
The A96L116 has a free running 8-bit Basic Interval Timer (BIT). This BIT generates the time base for Watchdog Timer counting and provides a basic interval timer interrupt (BITIFR).

The BIT of A96L116 features the followings:

- During Power-On, BIT gives a stable clock generation time.
- On exiting Stop mode, BIT gives a stable clock generation time.
- As a timer, BIT generates a timer interrupt.

8.1 Block Diagram

Figure 26. Basic Interval Timer in Block Diagram



8.2 Register Map

Table 13. Basic Interval Timer Register Map

Name	Address	Direction	Default	Description
BITCNT	8CH	R	00H	Basic Interval Timer Counter Register
BITCR	8BH	R/W	01H	Basic Interval Timer Control Register

8.3 Register Description

BITCNT (Basic Interval Timer Counter Register): 8CH

7	6	5	4	3	2	1	0
BITCNT7	BITCNT6	BITCNT5	BITCNT4	BITCNT3	BITCNT2	BITCNT1	BITCNT0
R	R	R	R	R	R	R	R

Initial value: 00H

BITCNT[7:0] BIT Counter

BITCR (Basic Interval Timer Control Register): 8BH

7	6	5	4	3	2	1	0
BITIFR	BITCK1	BITCK0	–	BCLR	BCK2	BCK1	BCK0
R/W	R/W	R/W	–	R/W	R/W	R/W	R/W

Initial value: 01H

BITIFR When BIT Interrupt occurs, this bit becomes ‘1’. For clearing bit, write ‘0’ to this bit or auto clear by INT_ACK signal. Writing “1” has no effect.

0 BIT interrupt no generation

1 BIT interrupt generation

BITCK[1:0] Select BIT clock source

BITCK1	BITCK0	Description
0	0	fx/4096
0	1	fx/1024
1	0	fx/128
1	1	fx/16

BCLR If this bit is written to ‘1’, BIT Counter is cleared to ‘0’

0 Free Running

1 Clear Counter

BCK[2:0] Select BIT overflow period

BCK2	BCK1	BCK0	Description
0	0	0	Bit 0 overflow (BIT Clock × 2)
0	0	1	Bit 1 overflow (BIT Clock × 4) (default)
0	1	0	Bit 2 overflow (BIT Clock × 8)
0	1	1	Bit 3 overflow (BIT Clock × 16)
1	0	0	Bit 4 overflow (BIT Clock × 32)
1	0	1	Bit 5 overflow (BIT Clock × 64)
1	1	0	Bit 6 overflow (BIT Clock × 128)
1	1	1	Bit 7 overflow (BIT Clock × 256)

9 Watchdog Timer (WDT)

Watchdog Timer (WDT) is used to rapidly detect CPU malfunctions such as endless looping caused by noise. In addition, it is used to resume the CPU in a normal state. Watchdog timer signal for malfunction detection can be used as either a CPU reset or an interrupt request. When the WDT is not being used for the detection of the CPU malfunctions, it can be used as a timer generating an interrupt at fixed intervals.

The WDT can be used in a free running 8-bit timer mode or in a watchdog timer mode by setting WDTRSON bit, which is WDTCR[6]. If '1' is written to WDTCR[5], WDT counter value is cleared and counts up. After 1 machine cycle, this bit is cleared to '0' automatically.

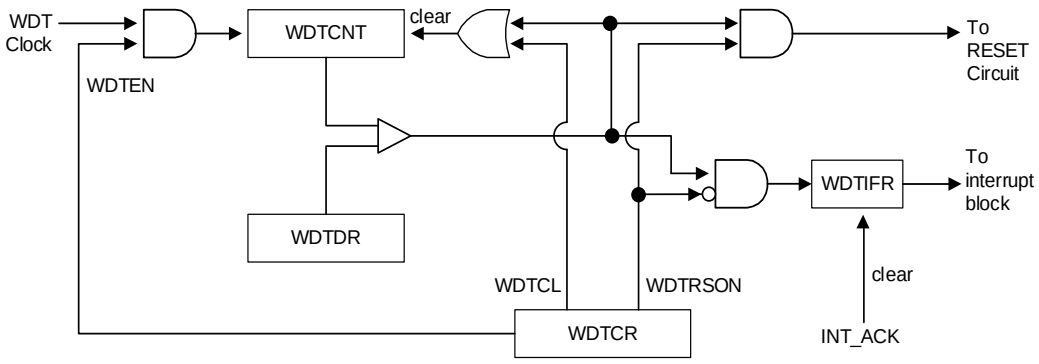
The WDT consists of an 8-bit binary counter and a watchdog timer data register. When the value of an 8-bit binary counter is equal to the 8 bits of WDTCNT, an interrupt request flag is generated. This can be used as a watchdog timer interrupt or a reset of CPU in accordance with a bit WDTRSON.

The input clock source of watchdog timer is BIT overflow and WDTRC. The interval of watchdog timer interrupt is decided by the BIT overflow period and WDTDR set value. Equation of the WDT interrupt interval is described in the followings:

- WDT Interrupt Interval = (BIT Interrupt Interval) × (WDTDR Value + 1)
- WDT Interrupt Interval = $2048/f_{LFIRC}$ × (WDTDR Value + 1) when LFIRC
- WDT Interrupt Interval = $256/f_{WDTRC}$ × (WDTDR Value + 1) when WDTRC

9.1 Block Diagram

Figure 27. Watchdog Timer in Block Diagram



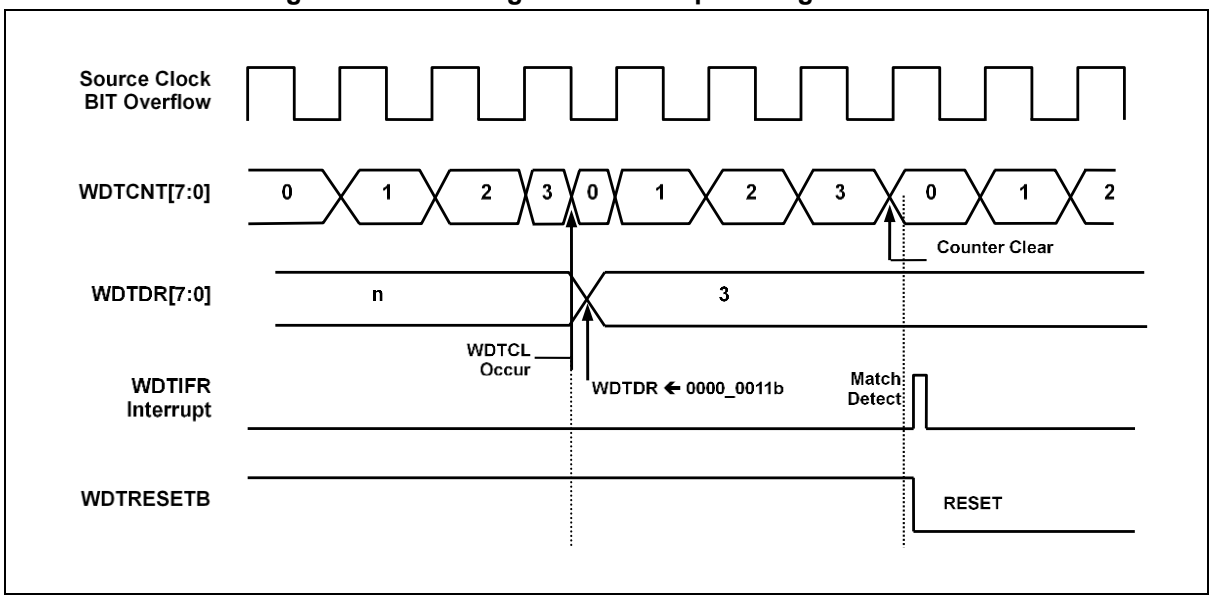
NOTE:

1. The watchdog timer clock depends on the WDTCK[1:0] bits in the WDTCR register. For detailed information of the WDTCR register, please refer to Watchdog timer register description later part in this chapter.

9.2 WDT Interrupt Timing Waveform

Figure 28 shows a timing diagram when a Watchdog Timer generates system reset signal and an interrupt signal.

Figure 28. Watchdog Timer Interrupt Timing Waveform



9.3 Register Map

Table 14. Watchdog Timer Register Map

Name	Address	Direction	Default	Description
WDTCNT	8EH	R	00H	Watchdog Timer Counter Register
WDTDR	8EH	W	FFH	Watchdog Timer Data Register
WDTCR	8DH	R/W	00H	Watchdog Timer Control Register

9.4 Register Description

WDTCNT (Watchdog Timer Counter Register: Read Case): 8EH

7	6	5	4	3	2	1	0
WDTCNT 7	WDTCNT 6	WDTCNT 5	WDTCNT 4	WDTCNT3	WDTCNT 2	WDTCNT 1	WDTCNT 0
R	R	R	R	R	R	R	R

Initial value: 00H

WDTCNT[7:0] WDT Counter

WDTDR (Watchdog Timer Data Register: Write Case): 8EH

7	6	5	4	3	2	1	0
WDTDR7	WDTDR6	WDTDR5	WDTDR4	WDTDR3	WDTDR2	WDTDR1	WDTDR0
W	W	W	W	W	W	W	W

Initial value: FFH

WDTDR[7:0] Set a period.
WDT Interrupt Interval is as follows:
(BIT Interrupt Interval) × (WDTDR Value + 1)
$256/f_{WDTRC}$ × (WDTDR Value + 1) when WDTRC
$2048/f_{LFIRC}$ × (WDTDR Value + 1) when LFIRC

NOTE:

1. Do not write “0” in the WDTDR register.

WDTCR (Watchdog Timer Control Register): 8DH

7	6	5	4	3	2	1	0
WDTEN	WDTRSON	WDTCL	–	–	WDTCK1	WDTCK0	WDTIFR
R/W	R/W	R/W	–	–	R/W	R/W	R/W

Initial value: 00H

WDTEN Control WDT Operation
0 Disable
1 Enable

WDTRSON Control WDT RESET Operation
0 Free Running 8-bit timer
1 Watchdog Timer RESET ON

WDTCL Clear WDT Counter
0 Free Run
1 Clear WDT Counter (auto clear after 1 Cycle)

WDTCK[1:0] Select WDT clock Selection Bits

WDTCK1	WDTCK0	Description
0	0	BIT overflow for WDT clock (WDTRC disable)
0	1	LFIRC for WDT clock (WDTRC disable)
1	0	WDTRC for WDT clock (WDTRC enable)
1	1	Not available (No effect)

WDTIFR When WDT Interrupt occurs, this bit becomes ‘1’. For clearing bit, write ‘0’ to this bit or auto clear by INT_ACK signal. Writing “1” has no effect.
0 WDT Interrupt no generation
1 WDT Interrupt generation

10 Real-Time Clock and Calendar (RTCC)

The Real-Time Clock and Calendar (RTCC) has a function for RTC (Real-Time Clock) and calendar operations.

The internal structure of the RTCC is implemented with the clock source select circuit, second/minute/hour/day/week/month/year counter circuits, alarm circuit, output select circuit, and error correction circuit.

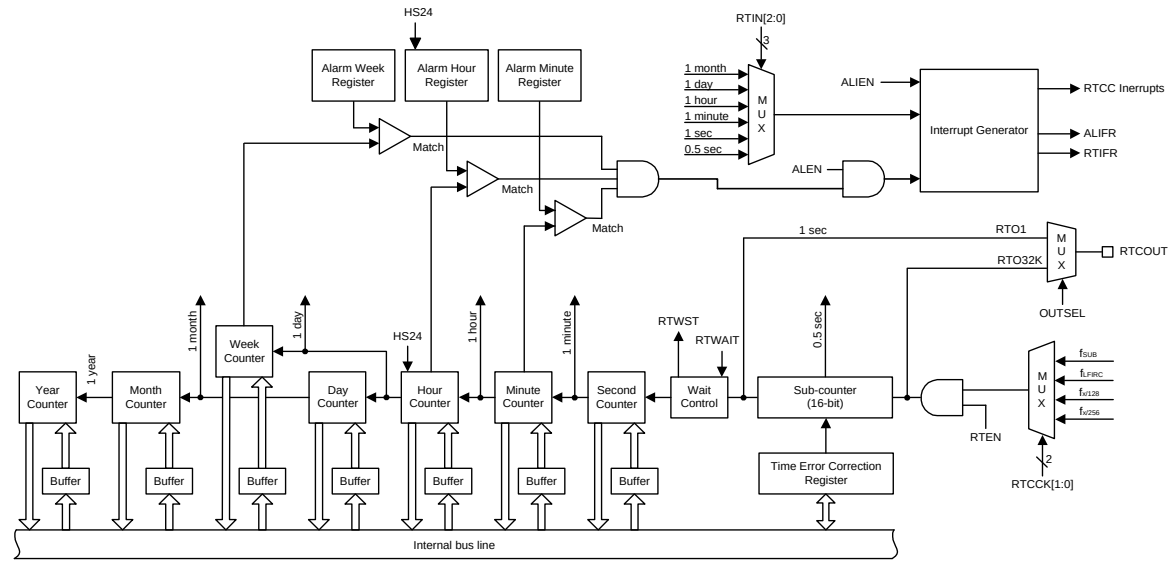
The RTCC is an independent BCD counter.

The main operations of the RTCC include the followings:

- Calendar counting 0.5 seconds, seconds, minutes, hours, days, weeks, months, and years up to the year 2099.
- Time error correction function
- Alarm function with interrupt
- Wake-up possibility from STOP mode

10.1 Block Diagram

Figure 29. RTCC Block Diagram



NOTE:

1. Time in the block diagram is based on 32.768 kHz.

10.2 Register Map

Table 15. Real Timer Clock and Calendar Register Map

Name	Address	Direction	Default	Description
RTCCRL	10B0H (XSFR)	R/W	00H	RTCC Control Low Register
RTCCRH	10B1H (XSFR)	R/W	00H	RTCC Control High Register
RTCSCTL	10B2H (XSFR)	R	00H	RTCC Sub-counter Low Register
RTCSCTH	10B3H (XSFR)	R	00H	RTCC Sub-counter High Register
RTCECR	10B4H (XSFR)	R/W	00H	RTCC Time Error Correction Register
RTCSEC	10B5H (XSFR)	R/W	00H	RTCC Second Counter Register
RTCMIN	10B6H (XSFR)	R/W	00H	RTCC Minute Counter Register
RTCHR	10B7H (XSFR)	R/W	12H	RTCC Hour Counter Register
RTCDAY	10B8H (XSFR)	R/W	01H	RTCC Day Counter Register
RTCWK	10B9H (XSFR)	R/W	00H	RTCC Week Counter Register
RTCMTH	10BAH (XSFR)	R/W	01H	RTCC Month Counter Register
RTCYR	10BBH (XSFR)	R/W	00H	RTCC Year Counter Register
RTCAMIN	10BCH (XSFR)	R/W	00H	RTCC Alarm Minute Register
RTCAHR	10BDH (XSFR)	R/W	12H	RTCC Alarm Hour Register
RTCAWK	10BEH (XSFR)	R/W	00H	RTCC Alarm Week Register

10.3 Register Description

RTCCRH (Real Timer Clock/Calendar Control High Register): 10B1H (XSFR)

7	6	5	4	3	2	1	0
RTEN	RTIN2	RTIN1	RTIN0	RTIFR	HS24	–	OUTSEL
R/W	R/W	R/W	R/W	R/W	R/W	–	R/W

Initial value: 00H

RTEN Control Real Timer Clock/Calendar

0	Disable
1	Enable

RTIN[2:0] RTCC Interrupt Interval Selection bits. NOTE 1

RTIN2	RTIN1	RTIN0	Description
0	0	0	Disable RTCC interval interrupt
0	0	1	Once per 0.5 sec
0	1	0	Once per 1 sec
0	1	1	Once per 1 min
1	0	0	Once per 1 hour
1	0	1	Once per 1 day
1	1	0	Once per 1 month
1	1	1	Not available, Value is not changed

RTIFR RTCC Interval Interrupt Flag bit. The flag is cleared only by writing ‘0’ to the bit. So, the flag should be cleared by software.

0	No request occurred
1	Request occurred.

HS24 12/24-hour System Selection bit. NOTE 2

0	12-hour system
1	24-hour system

OUTSEL RTCOUT Selection bit

0	RTO 1 (1 Hz)
1	RTO 32K (32 kHz)

RTCCRL (Real Timer Clock/Calendar Control Low Register): 10B0H (XSFR)

7	6	5	4	3	2	1	0
ALEN	ALIEN	ALIFR	–	RTCCK1	RTCCK0	RTWST	RTWAIT
R/W	R/W	R/W	–	R/W	R/W	R	R/W

Initial value: 00H

ALEN RTCC Alarm Match Operation Enable bit. NOTE 3
- 0 Disable RTCC alarm match operation
- 1 Enable RTCC alarm match operation

ALIEN RTCC Alarm Match Interrupt Enable bit.
- 0 Disable
- 1 Enable

ALIFR RTCC Alarm Match Interrupt Flag bit. The flag is cleared only by writing '0' to the bit. So, the flag should be cleared by software.
- 0 No request occurred
- 1 request occurred

RTCCK[1:0] Determine source clock

RTCCK1	RTCCK0	Description
0	0	f_{SUB}
0	1	f_{LFIRC}
1	0	fx/128
1	1	fx/256

RTWST RTCC Wait Status Flag bit. NOTE 4
- 0 Counter is operating
- 1 Mode to read/write counter value

RTWAIT RTCC Wait Control bit. NOTE 5
- 0 Set counter operation
- 1 Stop RTSEC to RTYEAR counters for read/write counter value

NOTES:

1. When changing the values of RTIN[2:0] while the counter operates (RTEN = 1), rewrite the values of RTIN[2:0] after disabling interrupt servicing RTCC interrupt by using the interrupt enable register 3(IE3). Furthermore, after rewriting the values of RTIN[2:0], enable the interrupt service after clearing the RTIFR flag.
2. Rewrite the HS24 value after setting RTWAIT (bit 0 of RTCCRL) to 1. If the HS24 value is changed, the values of the RTCC hour counter register (RTCHR) and RTCC alarm hour register (RTCAHR) change according to the specified time system. 'Value of RTCHR/RTCAHR by HS24 bit' Table shows the displayed time digits.
3. When setting a value to the ALEN bit while the counter operates (RTEN = 1) and ALIEN = 1, rewrite the ALEN bit after disabling interrupt servicing RTCC Interrupt by using the interrupt enable register 3(IE3). Furthermore, clear the ALIFR flag after rewriting the ALEN bit. When setting each alarm register (ALIEN flag of RTCCRL, the RTCAMIN register, the RTCAHR register, and the RTCAWK register), set match operation to be invalid ("0") for the ALEN bit.
4. This status flag indicates whether the setting of RTWAIT is valid. Before reading or writing the counter value, confirm that the value of this flag is 1.
5. This bit controls the operation of the counter. Be sure to write "1" to it to read or write the counter value. Because the RTCC sub-counter (RTCSCTH/L) continues operation, complete reading or writing of it in 1 second, and clear this bit back to 0. When RTWAIT = 1, it takes up to 2 clocks (RTCC clock) until the counter value can be read or written. If the RTCC sub-counter (RTCSCTH/L) overflows when RTWAIT = 1, it counts up after RTWAIT = 0. If the RTCC second counter register (RTCSEC) is written, however, it does not count up because RTCC sub-counter (RTCSCTH/L) is cleared.
6. fx – System clock frequency (where fx=4.19 MHz)
   f_{SUB} – Sub clock oscillator frequency (32.768 kHz)
   f_{LFIRC} – Low frequency IRC (40 kHz)

RTCSCTH (Real Timer Clock/Calendar Sub-counter High Register): 10B3H (XSFR)

7	6	5	4	3	2	1	0
RTCNT15	RTCNT14	RTCNT13	RTCNT12	RTCNT11	RTCNT10	RTCNT9	RTCNT8
R	R	R	R	R	R	R	R

Initial value: 00H

RTCNT[15:8] RTCC Sub-counter High Byte

RTCSCTL (Real Timer Clock/Calendar Sub-counter Low Register): 10B2H (XSFR)

7	6	5	4	3	2	1	0
RTCNT7	RTCNT6	RTCNT5	RTCNT4	RTCNT3	RTCNT2	RTCNT1	RTCNT0
R	R	R	R	R	R	R	R

Initial value: 00H

RTCNT[7:0] RTCC Sub-counter Low Byte

NOTE:

1. When a correction is made by using the RTCECR register, the value of RTCSCTH/L may become 8000H or more. The RTCSCTH/L is also cleared by writing the RTCC second counter register.
   The value read from this register is not guaranteed if it is read during operation, because a value that is changing is read.

RTCECR (Real Timer Clock/Calendar Time Error Correction Register): 10B4H (XSFR)

7	6	5	4	3	2	1	0
ECTM	ECSIGN	ECV5	ECV4	ECV3	ECV2	ECV1	ECV0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

ECTM Time Error Correction Timing Selection bit. NOTE 1

- 0 Corrects time error when the second digits are at 00H, 20H, or 40H. (every 20 seconds)
- 1 Corrects watch error only when the second digits are at 00H. (every 60 seconds)

ECSIGN Time Error Correction Data Sign bit. NOTE 2-4

- 0 Increases by {(ECV5, ECV4, ECV3, ECV2, ECV1, ECV0) –1} × 2.
- 1 Decreases by {(/ECV5, /ECV4, /ECV3, /ECV2, /ECV1, /ECV0) +1} × 2.

ECV[5:0] Time Error Correction Data bits.

NOTES:

1. Do not write to the RTC_ECR register at the following timing.
   - When ECTM = 0 is set: For a period of SEC = 00H, 20H, 40H
   - When ECTM = 1 is set: For a period of SEC = 00H
2. When (ECSIGN, ECV5, ECV4, ECV3, ECV2, ECV1, ECV0) = (n, 0, 0, 0, 0, 0, n), the watch error is not corrected. (Where n = 0 or 1.)
3. /ECV5 to /ECV0 are the inverted values of the corresponding bits (000011 when 111100).
4. Range of correction value: (when ECSIGN = 0) 2, 4, 6, 8, … , 120, 122, 124
   (when ECSIGN = 1) –2, –4, –6, –8, … , –120, –122, –124

RTCSEC (Real Timer Clock/Calendar Second Counter Register): 10B5H (XSFR)

7	6	5	4	3	2	1	0
–	RSEC6	RSEC5	RSEC4	RSEC3	RSEC2	RSEC1	RSEC0
–	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

RSEC[6:0] RTCC Second Counter bits

NOTE:

1. The RTCSEC register takes a value of 0 to 59 (decimal) and indicates the count value of seconds.
   It counts up when the RTCC sub-counter overflows.
   Set a decimal value of 00 to 59 to this register in BCD code. If a value outside this range is tried to be written in the register, the value is ignored.

RTCMIN (Real Timer Clock/Calendar Minute Counter Register): 10B6H (XSFR)

7	6	5	4	3	2	1	0
–	RMIN6	RMIN5	RMIN4	RMIN3	RMIN2	RMIN1	RMIN0
–	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

RMIN[6:0] RTCC Minute Counter bits

NOTE:

1. The RTCMIN register takes a value of 0 to 59 (decimal) and indicates the count value of minutes. It counts up when the RTCC second counter register overflows.
   Even if the RTCC second counter register overflows while this register is being written, this register ignores the overflow and is set to the value written.
   Set a decimal value of 00 to 59 to this register in BCD code. If a value outside this range is tried to be written in the register, the value is ignored.

RTCHR (Real Timer Clock/Calendar Hour Counter Register): 10B7H (XSFR)

7	6	5	4	3	2	1	0
–	–	RHOUR5	RHOUR4	RHOUR3	RHOUR2	RHOUR1	RHOUR0
–	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 12H

RHOUR[5:0] RTCC Hour Counter bits

NOTE:

1. The RTCHR register takes a value of 00 to 23 or 01 to 12, 21 to 32 (decimal) and indicates the count value of hours.
   It counts up when the RTCC minute counter register overflows.
   Even if the RTCC minute counter register overflows while this register is being written, this register ignores the overflow and is set to the value written. Specify a decimal value of 00 to 23, 01 to 12, or 21 to 32 by using BCD code according to the time system specified using HS24 bit of RTCCRH(RTCC control high register). If a value outside this range is tried to be written in the register, the value is ignored.
   RHOUR5 bit of RTCHR indicates AM(0)/PM(1) if HS24(RTCCRH[2])= 0 (if the 12-hour system is selected).

Table 16. Value of RTCHR/RTCAHR by HS24 Bit

24-Hour Display (HS24 bit=1)		12-Hour Display (HS24 bit=0)	
Time	**RTCHR Register RTCAHR Register**	**Time**	**RTCHR Register RTCAHR Register**
0	00H	0 a.m.	12H
1	01H	1 a.m.	01H
2	02H	2 a.m.	02H
3	03H	3 a.m.	03H
4	04H	4 a.m.	04H
5	05H	5 a.m.	05H
6	06H	6 a.m.	06H
7	07H	7 a.m.	07H
8	08H	8 a.m.	08H
9	09H	9 a.m.	09H
10	10H	10 a.m.	10H
11	11H	11 a.m.	11H
12	12H	0 p.m.	32H
13	13H	1 p.m.	21H
14	14H	2 p.m.	22H
15	15H	3 p.m.	23H
16	16H	4 p.m.	24H
17	17H	5 p.m.	25H
18	18H	6 p.m.	26H
19	19H	7 p.m.	27H
20	20H	8 p.m.	28H
21	21H	9 p.m.	29H
22	22H	10 p.m.	30H
23	23H	11 p.m.	31H

RTCDAY (Real Timer Clock/Calendar Day Counter Register): 10B8H (XSFR)

7	6	5	4	3	2	1	0
–	–	RDAY5	RDAY4	RDAY3	RDAY2	RDAY1	RDAY0
–	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 01H

RDAY[5:0] RTCC Day Counter bits

NOTE:

1. The RTCDAY register takes a value of 1 to 31 (decimal) and indicates the count value of days. It counts up when the RTCC hour counter register overflows.
   Even if the RTCC hour counter register overflows while this register is being written, this register ignores the overflow and is set to the value written.
   Set a decimal value of 01 to 31 to this register in BCD code. If a value outside this range is tried to be written in the register, the value is ignored.
   The RTCDAY register counts as follows
   - 01 to 31 (January, March, May, July, August, October, December)
   - 01 to 30 (April, June, September, November)
   - 01 to 29 (February of leap year)
   - 01 to 28 (February of normal year).

RTCWK (Real Timer Clock/Calendar Week Counter Register): 10B9H (XSFR)

7	6	5	4	3	2	1	0
–	–	–	–	–	RWEEK2	RWEEK1	RWEEK0
–	–	–	–	–	R/W	R/W	R/W

Initial value: 00H

RWEEK[2:0] RTCC Week Counter bits

RWEEK2	RWEEK1	RWEEK0	Description
0	0	0	Sunday
0	0	1	Monday
0	1	0	Tuesday
0	1	1	Wednesday
1	0	0	Thursday
1	0	1	Friday
1	1	0	Saturday
1	1	1	Not available, Value is not changed

NOTE:

1. The RTCWK register takes a value of 0 to 6 (decimal) and indicates the count value of weekdays. It counts up in synchronization with the RTCC day counter register.
   Set a decimal value of 00 to 06 to this register in BCD code. If a value outside this range is tried to be written in the register, the value is ignored.
   Values corresponding to the month count register and day count register are not automatically stored to the RTCC week counter register.

RTCMTH (Real Timer Clock/Calendar Month Counter Register): 10BAH (XSFR)

7	6	5	4	3	2	1	0
–	–	–	RMONTH4	RMONTH3	RMONTH2	RMONTH1	RMONTH0
–	–	–	R/W	R/W	R/W	R/W	R/W

Initial value: 01H

RMONTH[4:0] RTCC Month Counter bits

NOTE:

1. The RTCMTH register takes a value of 1 to 12 (decimal) and indicates the count value of months. It counts up when the RTCC day counter register overflows.
   Even if the RTCC day counter register overflows while this register is being written, this register ignores the overflow and is set to the value written.
   Set a decimal value of 01 to 12 to this register in BCD code. If a value outside this range is tried to be written in the register, the value is ignored.

RTCYR (Real Timer Clock/Calendar Year Counter Register): 10BBH (XSFR)

7	6	5	4	3	2	1	0
RYEAR7	RYEAR6	RYEAR5	RYEAR4	RYEAR3	RYEAR2	RYEAR1	RYEAR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

RYEAR[7:0] RTCC Year Counter bits

NOTE:

1. The RTCYR register takes a value of 0 to 99 (decimal) and indicates the count value of years. It counts up when the RTCC month counter register overflows.
   Values 00, 04, 08, …, 92, and 96 indicate a leap year. Even if the RTCC month counter register overflows while this register is being written, this register ignores the overflow and is set to the value written.
   Set a decimal value of 00 to 99 to this register in BCD code. If a value outside this range is tried to be written in the register, the value is ignored.

RTCAMIN (Real Timer Clock/Calendar Alarm Minute Counter Register): 10BCH (XSFR)

7	6	5	4	3	2	1	0
–	AMIN6	AMIN5	AMIN4	AMIN3	AMIN2	AMIN1	AMIN0
–	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

AMIN[6:0] RTCC Alarm Minute Counter bits

NOTE:

1. This register is used to set minutes of alarm. Set a decimal value of 00 to 59 to this register in BCD code. If a value outside this range is tried to be written in the register, the value is ignored.

RTCAHR (Real Timer Clock/Calendar Alram Hour Counter Register): 10BDH (XSFR)

7	6	5	4	3	2	1	0
–	–	AHOUR5	AHOUR4	AHOUR3	AHOUR2	AHOUR1	AHOUR0
–	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 12H

AHOUR[5:0] RTCC Alarm Hour Counter bits

NOTE:

1. This register is used to set hours of alarm. Specify a decimal value of 00 to 23, 01 to 12, or 21 to 32 by using BCD code according to the time system specified using HS24 bit of RTCCRH(RTCC control high register). If a value outside this range is tried to be written in the register, the value is ignored.
   AHOUR5 bit of RTCAHR indicates AM(0)/PM(1) if HS24 = 0 (if the 12-hour system is selected).

RTCAWK (Real Timer Clock/Calendar Alarm Week Counter Register): 10BEH (XSFR)

7	6	5	4	3	2	1	0
–	AWEEK6	AWEEK5	AWEEK4	AWEEK3	AWEEK2	AWEEK1	AWEEK0
–	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

AWEEK6 Saturday Alarm Setting bit.
- 0 Disable Saturday alarm
- 1 Enable Saturday alarm

AWEEK5 Friday Alarm Setting bit.
- 0 Disable Friday alarm
- 1 Enable Friday alarm

AWEEK4 Thursday Alarm Setting bit.
- 0 Disable Thursday alarm
- 1 Enable Thursday alarm

AWEEK3 Wednesday Alarm Setting bit.
- 0 Disable Wednesday alarm
- 1 Enable Wednesday alarm

AWEEK2 Tuesday Alarm Setting bit.
- 0 Disable Tuesday alarm
- 1 Enable Tuesday alarm

AWEEK1 Monday Alarm Setting bit.
- 0 Disable Monday alarm
- 1 Enable Monday alarm

AWEEK0 Sunday Alarm Setting bit.
- 0 Disable Sunday alarm
- 1 Enable Sunday alarm

11 TIMER 0

A 16-bit timer 0 incorporates a multiplexer and eight registers such as timer 0 A data register high/low, timer 0 B data register high/low, timer 0 capture data register high/low, and timer 0 control register high/low (T0ADRH, T0ADRL, T0BDRH, T0BDRL, T0CAPH, T0CAPL, T0CRH, T0CRL).

TIMER 0 operates in one of four operating modes:

- 16-bit capture mode
- 16-bit timer/counter mode
- 16-bit PPG output mode (one-shot mode)
- 16-bit PPG output mode (repeat mode)

Specifically in capture mode, data is captured into input capture data register (T0CAPH/T0CAPL) by EINT10, f_{SUB}, or f_{LFIRC}. Timer 0 outputs the comparison result between counter and data register through T0O port in timer/counter mode. Timer 0 outputs PWM wave form through PWM0O port in the PPG mode).

A timer/counter 0 uses an internal clock or an external clock (EC0, f_{SUB}, f_{LFIRC}) as an input clock source. The clock sources are listed below, and one is selected by clock selection logic which is controlled by clock selection bits (T0CK[2:0]).

- Timer 0 clock sources: f_X/1, 2, 4, 8, 16, f_{SUB}, f_{LFIRC} and EC0

Table 17. TIMER 0 Operating Modes

T0EN	P0FSRL[4]	T0MS[1:0]	T0CK[2:0]	Timer 0
1	1	00	XXX	16 Bit Timer/Counter Mode
1	0	01	XXX	16 Bit Capture Mode
1	1	10	XXX	16 Bit PPG Mode(One-shot mode)
1	1	11	XXX	16 Bit PPG Mode (Repeat mode)

11.1 16-bit Timer/Counter Mode

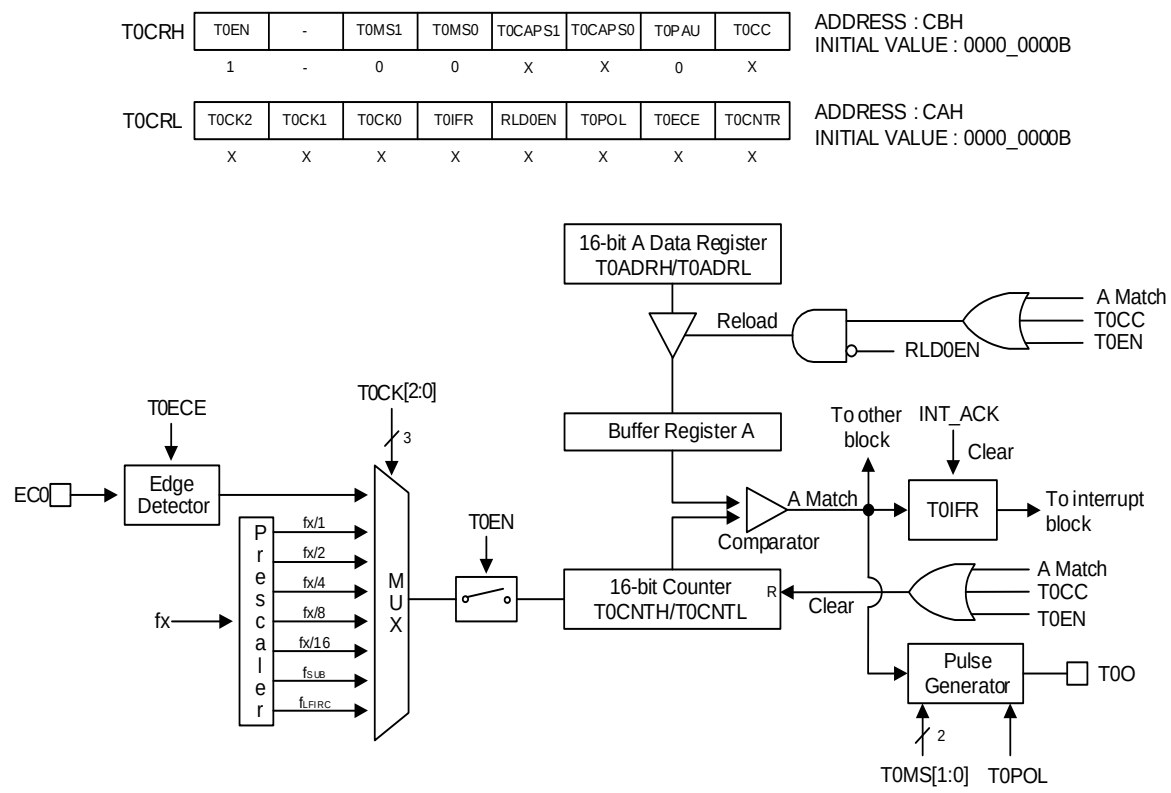
16-bit timer/counter mode is selected by control register as shown in Figure 30. This figure shows that a 16-bit timer has a counter and data registers. Counter registers have increasing values by internal or external clock input. TIMER 0 can use the input clock with one of 1, 2, 4, 8, 16, f_{SUB}, f_{LFIRC} and EC0 prescaler division rates (T0CK[2:0]).

When the value of T0CNTH, T0CNTL and the value of T0ADRH, T0ADRL are identical to each other in Timer 0, a match signal is generated, and the interrupt of Timer 0 occurs.

The T0CNTH, T0CNTL value is automatically cleared by match signal. It can be cleared by software (T0CC) too.

The external clock (EC0) counts up the timer at the rising edge. If the EC0 is selected as a clock source by T0CK[2:0], EC0 port should be set to the EC0 function by P0FSRH[0] bit.

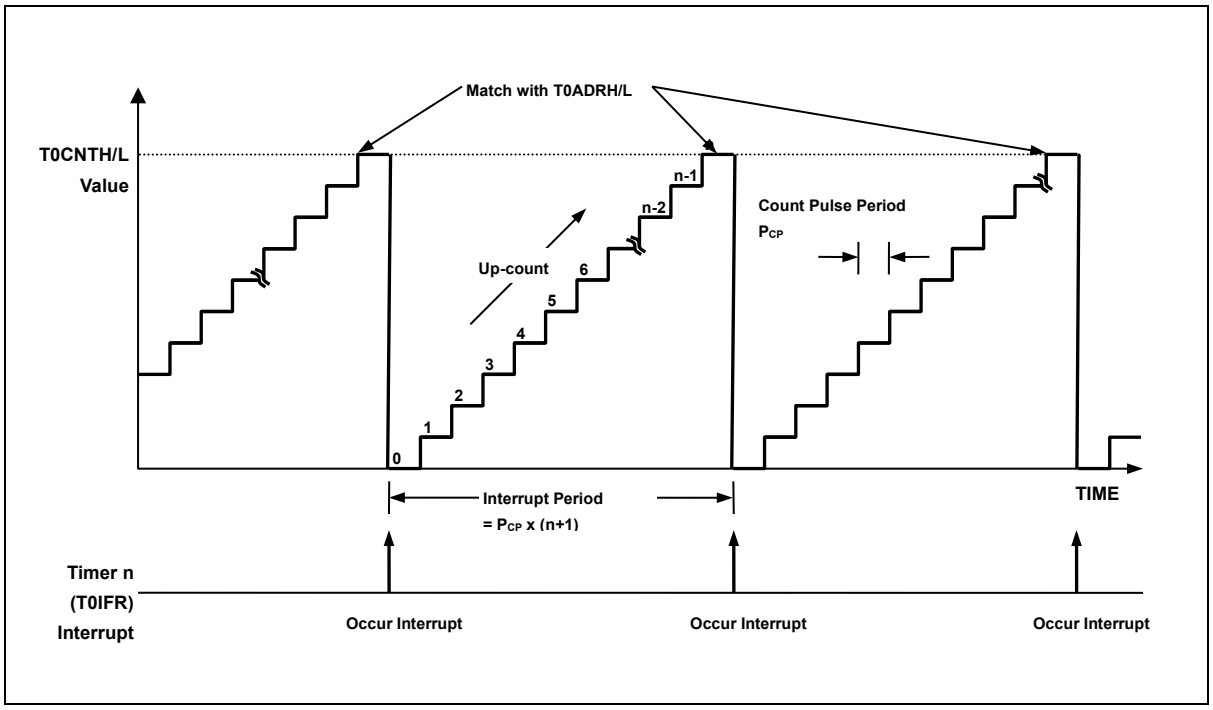
Figure 30. 16-bit Timer/Counter Mode of TIMER 0



NOTE:

1. RLD0EN bit must be '0b' when the T0EN bit is changed from '0b' to '1b'.

Figure 31. 16-bit Timer/Counter 0 Interrupt Example

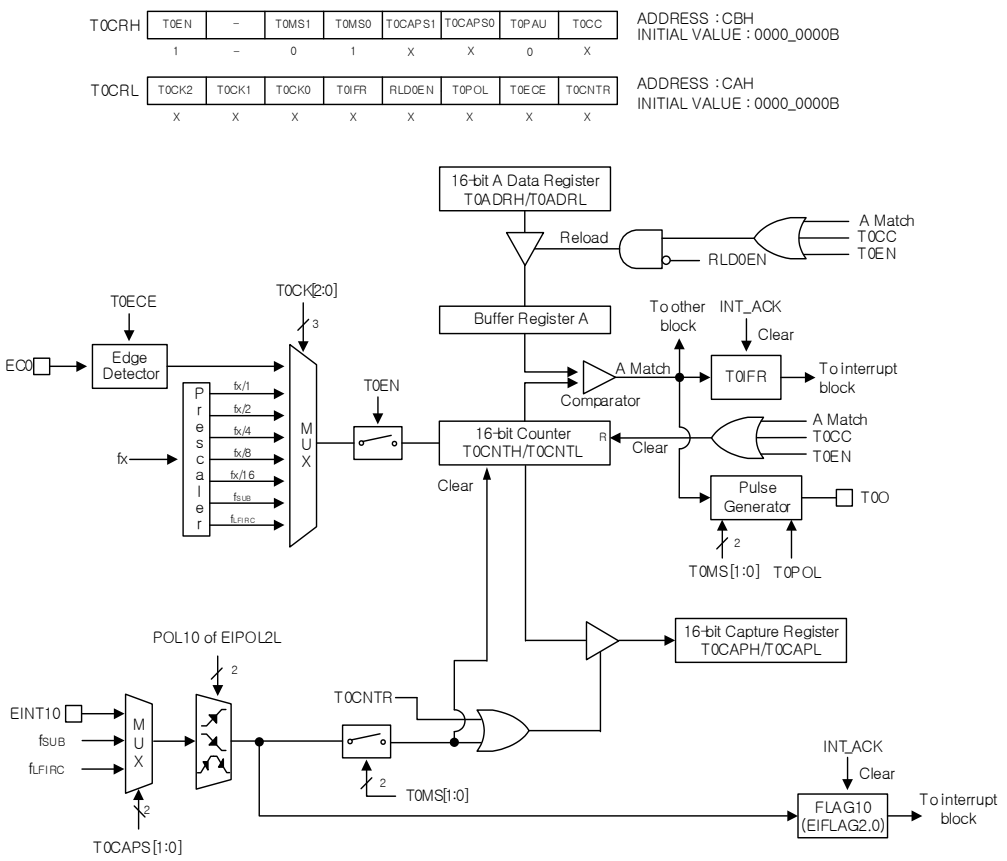


11.2 16-bit Capture Mode

16-bit timer 0 capture mode is set by configuring T0MS[1:0] as '01'. It uses an internal/external clock as a clock source. Basically, the 16-bit timer 0 capture mode has the same function as the 16-bit timer/counter mode, and the interrupt occurs when T0CNTH/T0CNTL is equal to T0ADRH/T0ADRL. The T0CNTH, T0CNTL values are automatically cleared by match signal. It can be cleared by software (T0CC) too.

A timer interrupt in capture mode is extremely useful when the pulse width of captured signal is wider than the maximum period of timer. Capture result is loaded into T0CAPH/L. According to T0CAPS[1:0] registers settings, the EINT10, f_{SUB}, f_{LFIRC} can be selected.

Figure 32. 16-bit Capture Mode of TIMER 0



NOTE:

1. RLD0EN bit must be '0b' when T0EN bit is changed from '0' to '1'.

Figure 33. Input Capture Mode Operation of TIMER 0

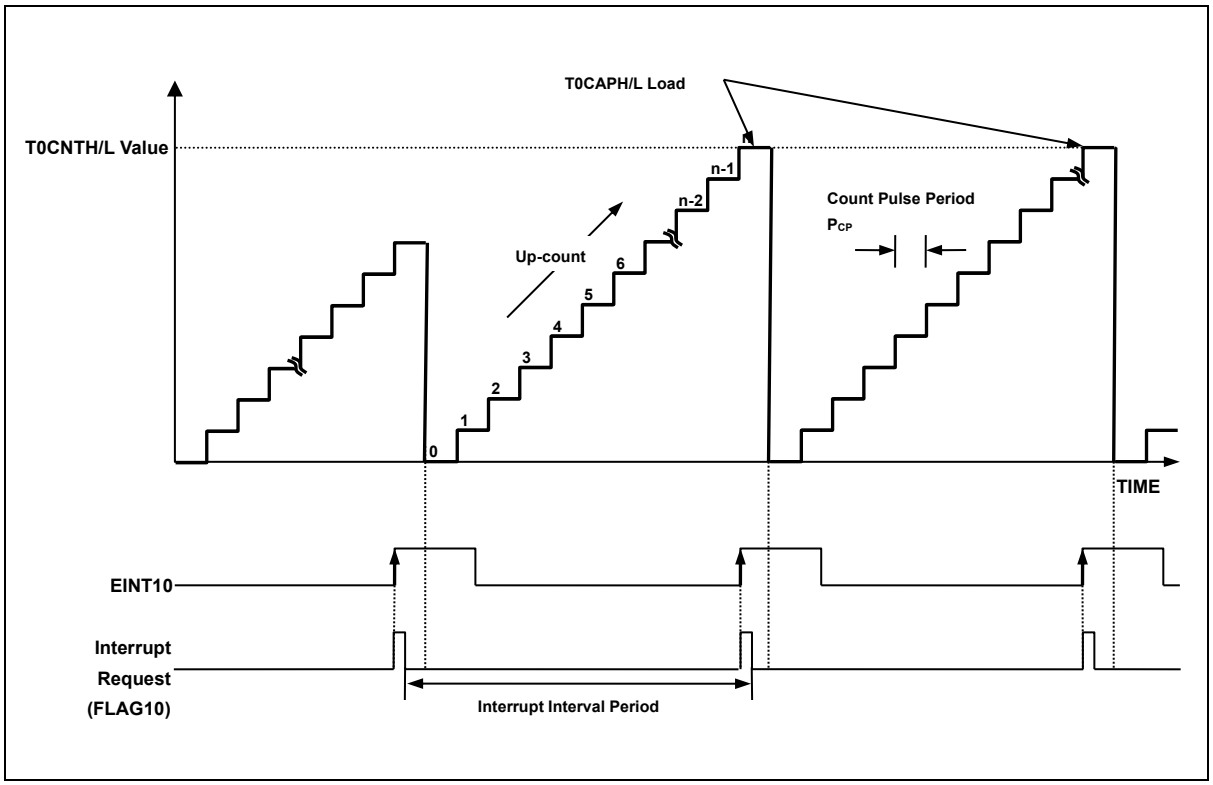
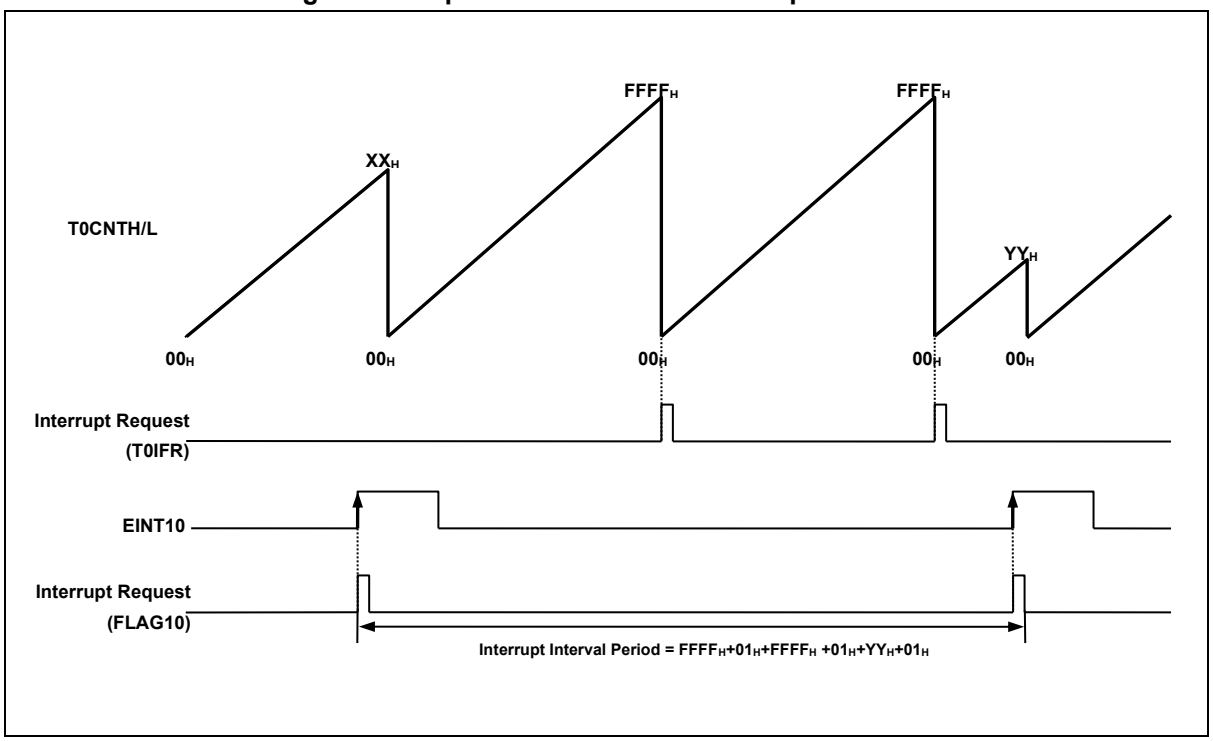


Figure 34. Express Timer Overflow in Capture Mode

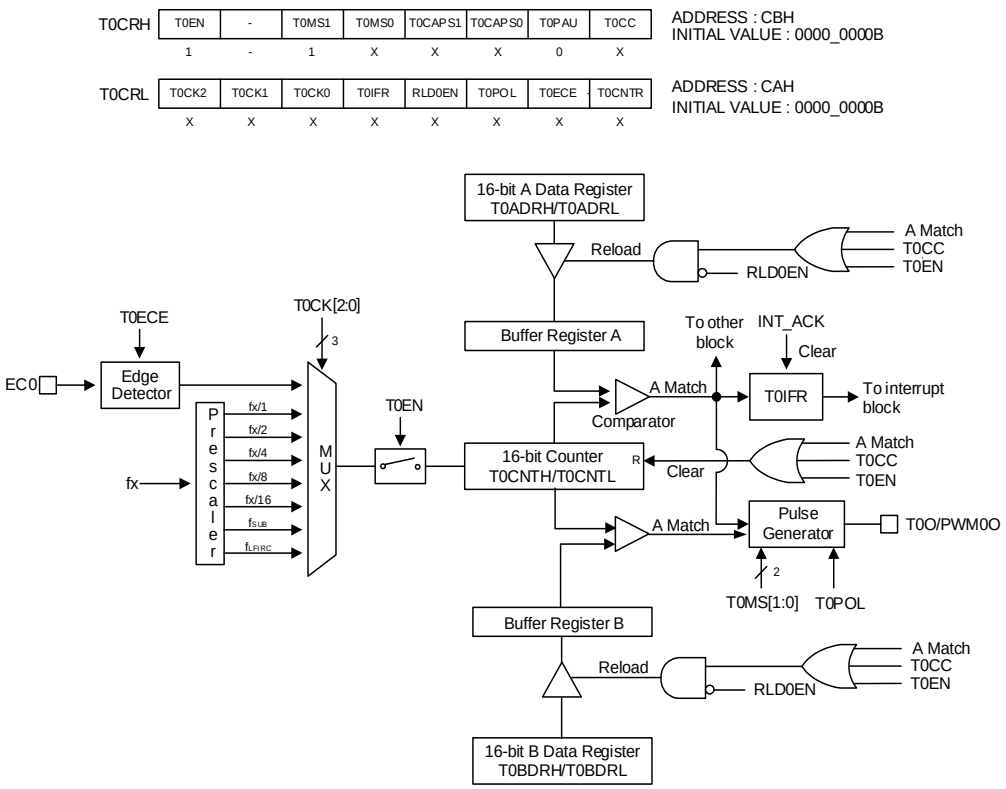


11.3 16-bit PPG Mode

TIMER 0 has a PPG (Programmable Pulse Generation) function. In PPG mode, T0O/PWM0O pin outputs up to 16-bit resolution PWM output.

For this function, T0O/PWM0O pin must be configured as a PWM output by setting P0FSRL[4] to '1'(T0). Period of the PWM output is determined by T0ADRH/T0ADRL, and duty of the PWM output is determined by T0BDRH/T0BDRL.

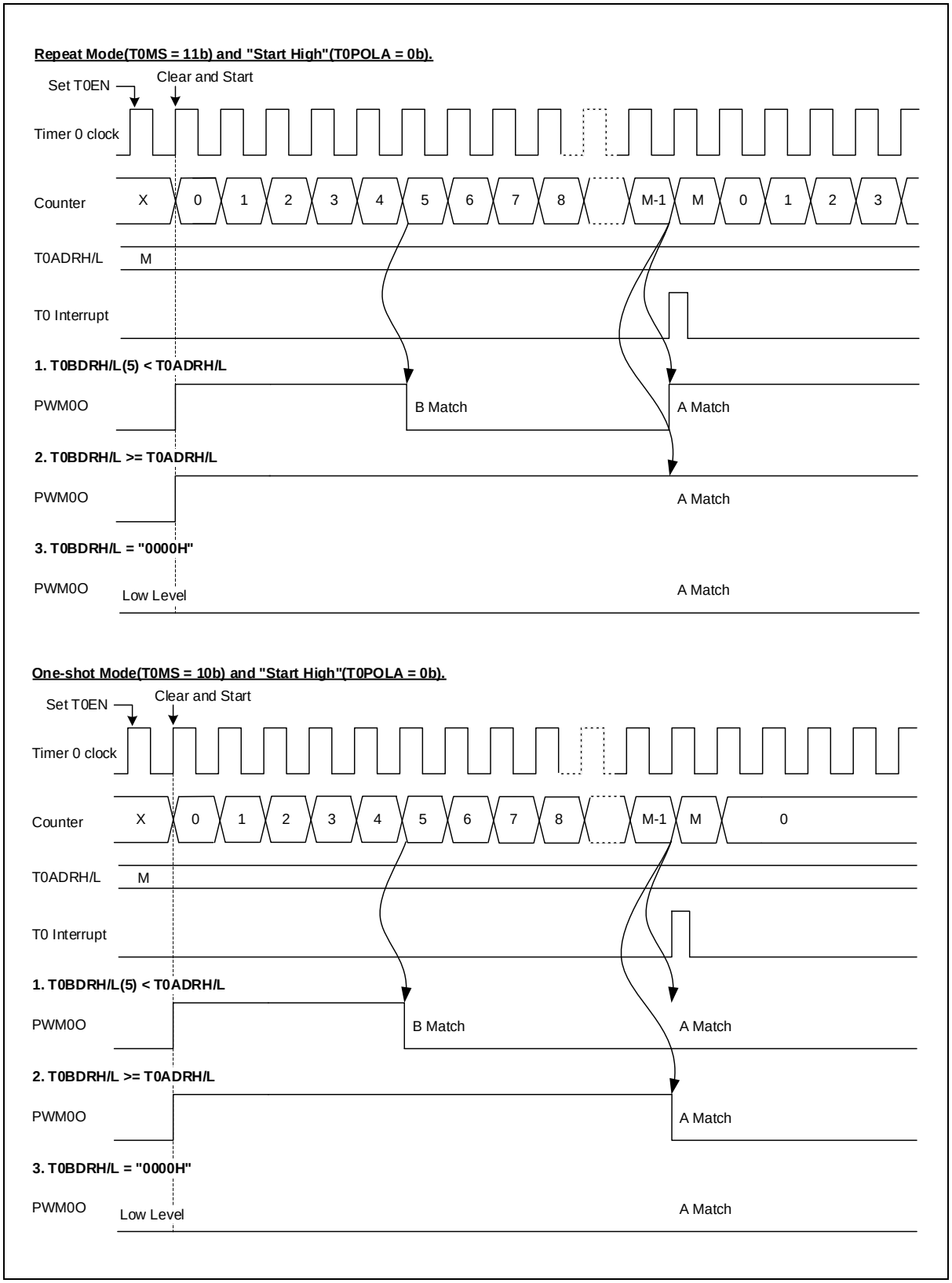
Figure 35. 16-bit PPG Mode of TIMER 0



NOTES:

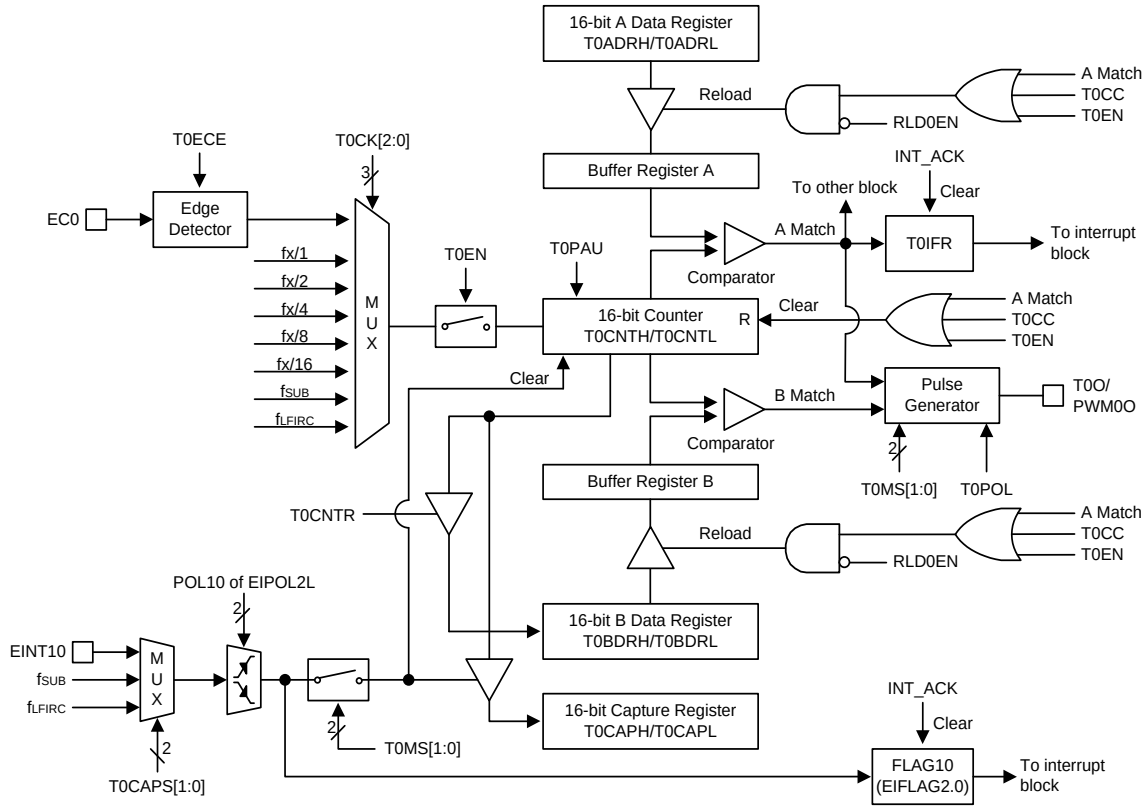
1. T0EN is automatically cleared to logic '0' after one pulse is generated at a PPG one-shot mode.
2. RLD0EN bit must be '0b' when T0EN bit is changed from '0' to '1'.

Figure 36. 16-bit PPG Mode Timing Chart of TIMER 0



11.4 Block Diagram

Figure 37. 16-bit Timer 0 in Block Diagram



NOTES:

1. T0EN is automatically cleared to logic ‘0’ after one pulse is generated at a PPG one-shot mode.
2. RLD0EN bit must be ‘0b’ when T0EN bit is changed from ‘0’ to ‘1’.

11.5 Register Map

Table 18. TIMER 0 Register Map

Name	Address	Direction	Default	Description
T0CRH	CBH	R/W	00H	Timer 0 Control High Register
T0CRL	CAH	R/W	00H	Timer 0 Control Low Register
T0ADRH	CDH	R/W	FFH	Timer 0 A Data High Register
T0ADRL	CCH	R/W	FFH	Timer 0 A Data Low Register
T0BDRH	CFH	R/W	FFH	Timer 0 B Data High Register
T0BDRL	CEH	R/W	FFH	Timer 0 B Data Low Register
T0CAPH	1007H(XSFR)	R	00H	Timer 0 Capture Data High Register
T0CAPL	1006H(XSFR)	R	00H	Timer 0 Capture Data Low Register

11.6 Timer/Counter 0 Register Description

T0ADRH (Timer 0 A data High Register): CDH

7	6	5	4	3	2	1	0
T0ADRH7	T0ADRH6	T0ADRH5	T0ADRH4	T0ADRH3	T0ADRH2	T0ADRH1	T0ADRH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

T0ADRH[7:0] T0 A Data High Byte

T0ADRL (Timer 0 A Data Low Register): CCH

7	6	5	4	3	2	1	0
T0ADRL7	T0ADRL6	T0ADRL5	T0ADRL4	T0ADRL3	T0ADRL2	T0ADRL1	T0ADRL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

T0ADRL[7:0] T0 A Data Low Byte

NOTE:

1. Do not write “0000H” in the T0ADRH/T0ADRL register when PPG mode

T0BDRH (Timer 0 B Data High Register): CFH

7	6	5	4	3	2	1	0
T0BDRH7	T0BDRH6	T0BDRH5	T0BDRH4	T0BDRH3	T0BDRH2	T0BDRH1	T0BDRH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

T0BDRH[7:0] T0 B Data High Byte

T0BDRL (Timer 0 B Data Low Register): CEH

7	6	5	4	3	2	1	0
T0BDRL7	T0BDRL6	T0BDRL5	T0BDRL4	T0BDRL3	T0BDRL2	T0BDRL1	T0BDRL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

T0BDRL[7:0]	T0 B Data Low Byte

T0CAPH (Timer 0 Capture Data High Register): 1007H (XSFR)

7	6	5	4	3	2	1	0
T0CAPH7	T0CAPH6	T0CAPH5	T0CAPH4	T0CAPH3	T0CAPH2	T0CAPH1	T0CAPH0
R	R	R	R	R	R	R	R

Initial value: 00H

T0CAPH[7:0]	T0 Capture Data High Byte

T0CAPL (Timer 0 Capture Data Low Register): 1006H (XSFR)

7	6	5	4	3	2	1	0
T0CAPL7	T0CAPL6	T0CAPL5	T0CAPL4	T0CAPL3	T0CAPL2	T0CAPL1	T0CAPL0
R	R	R	R	R	R	R	R

Initial value: 00H

T0CAPL[7:0]	T0 Capture Data Low Byte

T0CRH (Timer 0 Control High Register): CBH

7	6	5	4	3	2	1	0
T0EN	–	T0MS1	T0MS0	T0CAPS1	T0CAPS0	T0PAU	T0CC
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

T0EN Control Timer 0

0	Timer 0 disable
1	Timer 0 enable (counter clear and start)

T0MS[1:0] Control Timer 0 Operation Mode

T0MS1	T0MS0	Description
0	0	Timer/counter mode (T0O: toggle at A match)
0	1	Capture mode (The A match interrupt can occur)
1	0	PPG one-shot mode (PWM0O)
1	1	PPG repeat mode (PWM0O)

T0CAPS[1:0] Timer 0 Capture Signal Selection bits

T0CAPS1	T0CAPS0	Description
0	0	Select EINT10 for capture signal
0	1	Select f_{SUB} for capture signal
1	0	Select f_{LFIRC} for capture signal
1	1	Not available

T0PAU Timer 0 Counter Temporary Pause Control

0	Continue counting
1	Temporary pause

T0CC Clear Timer 0 Counter

0	No effect
1	Clear the Timer n counter (when write, automatically cleared “0” after being cleared counter)

T0CRL (Timer 0 Control Low Register): CAH

7	6	5	4	3	2	1	0
T0CK2	T0CK1	T0CK0	T0IFR	RLD0EN	T0POL	T0ECE	T0CNTR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

T0CK[2:0] Select Timer 0 clock source. fx is main system clock frequency

T0CK2	T0CK1	T0CK0	Description
0	0	0	fx/16
0	0	1	fx/8
0	1	0	fx/4
0	1	1	fx/2
1	0	0	fx/1
1	0	1	f_{SUB}
1	1	0	f_{LFIRC}
1	1	1	External clock (EC0)

T0IFR When T0 Interrupt occurs, this bit becomes ‘1’. For clearing bit, write ‘0’ to this bit or auto clear by INT_ACK signal. Writing “1” has no effect.

0 T0 Interrupt no generation

1 T0 Interrupt generation

RLD0EN Control Timer 0 Reload Signal

0 Enable Timer 0 reload signal

1 Disable Timer 0 reload signal

T0POL T0O/PWM0O Polarity Selection

0 Start High (T0O/PWM0O is low level at disable)

1 Start Low (T0O/PWM0O is high level at disable)

T0ECE Timer 0 External Clock Edge Selection

0 External clock falling edge

1 External clock rising edge

T0CNTR Timer 0 Counter Read Control

0 No effect

1 Load the counter value to the B data register (When write, automatically cleared “0” after being loaded)

12 TIMER 1/2

A 16-bit timer 1/2 incorporates a multiplexer and eight registers such as timer 1/2 A data register high/low, timer 1/2 B data register high/low, timer 1/2 capture data register high/low, and timer 1/2 control register high/low (TnADRH, TnADRL, TnBDRH, TnBDRL, TnCAPH, TnCAPL, TnCRH, TnCRL).

TIMER 1/2 operates in one of four operating modes:

- 16-bit capture mode
- 16-bit timer/counter mode
- 16-bit PPG output mode (One-shot mode)
- 16-bit PPG output mode (Repeat mode)

Specifically in capture mode, data is captured into capture data register (TnCAPH/TnCAPL) by EINT1n. Timer 1/2 outputs the comparison result between counter and data register through TnO port in timer/counter mode. Timer 1/2 outputs PWM wave form through PWMnO port in the PPG mode).

A timer/counter 1/2 uses an internal clock or an external clock (ECn) as an input clock source. The clock sources are listed below, and one is selected by clock selection logic which is controlled by clock selection bits (TnCK[2:0]).

- Timer 1/2 clock sources: f_X/1, 2, 4, 8, 64, 512, 2048 and Ecn

Table 19. TIMER 1/2 Operating Modes

TnEN	P0FSRL[6](T1)/ P1FSRL[6](T2)	TnMS[1:0]	TnCK[2:0]	Timer n
1	1/0	00	XXX	16 Bit Timer/Counter Mode
1	0/0	01	XXX	16 Bit Capture Mode
1	1/0	10	XXX	16 Bit PPG Mode(One-shot mode)
1	1/0	11	XXX	16 Bit PPG Mode (Repeat mode)

12.1 16-bit Timer/Counter Mode

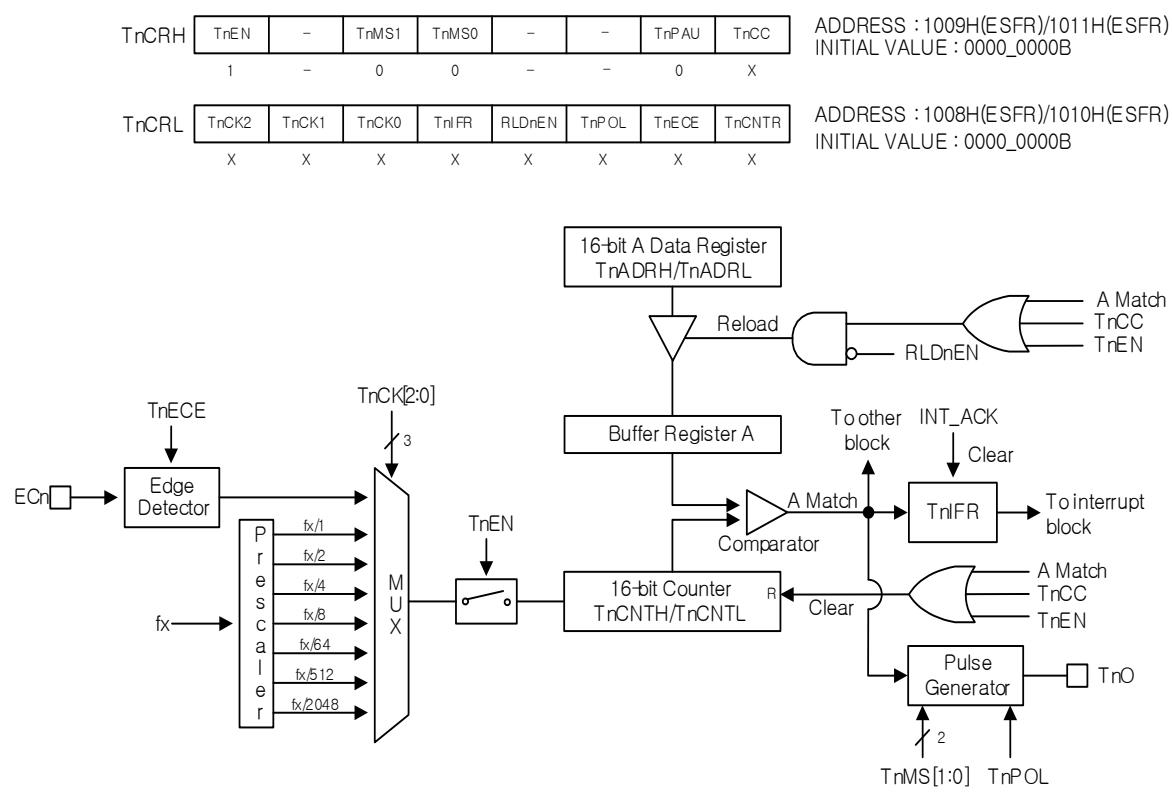
16-bit timer/counter mode is selected by control register as shown in Figure 38. This figure shows that a 16-bit timer has a counter and data registers. Counter registers have increasing values by internal or external clock input. TIMER 1/2 can use the input clock with one of 1, 2, 4, 8, 64, 512 and 2048 prescaler division rates (TnCK[2:0]).

When the value of TnCNTH, TnCNTL and the value of TnADRH, TnADRL are identical to each other in Timer 1/2, a match signal is generated, and the interrupt of Timer 1/2 occurs.

The TnCNTH, TnCNTL value is automatically cleared by match signal. It can be cleared by software (TnCC) too.

The external clock (ECn) counts up the timer at the rising edge. If the ECn is selected as a clock source by TnCK[2:0], ECn port should be set to the ECn function by P0FSRH[3:2]/P2FSRL[0] bits.

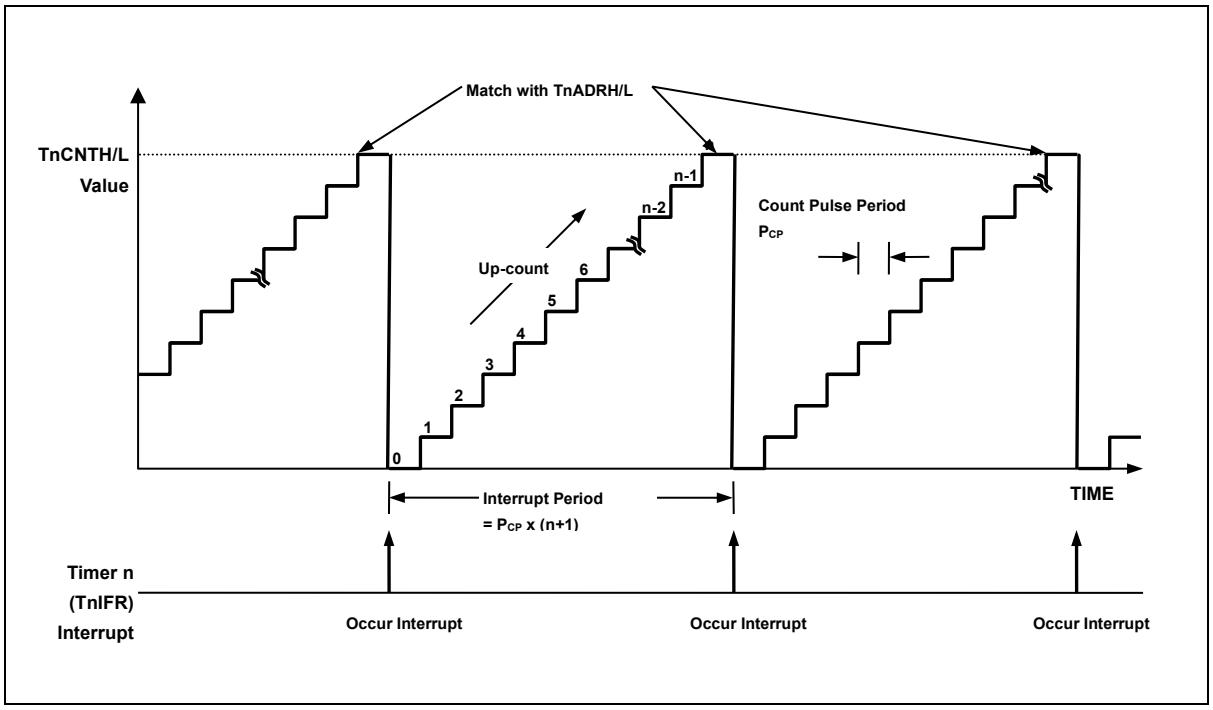
Figure 38. 16-bit Timer/Counter Mode of TIMER 1/2



NOTE:

1. RLDnEN bit must be '0' when the TnEN bit is changed from '0b' to '1b'.

Figure 39. 16-bit Timer/Counter 1/2 Interrupt Example

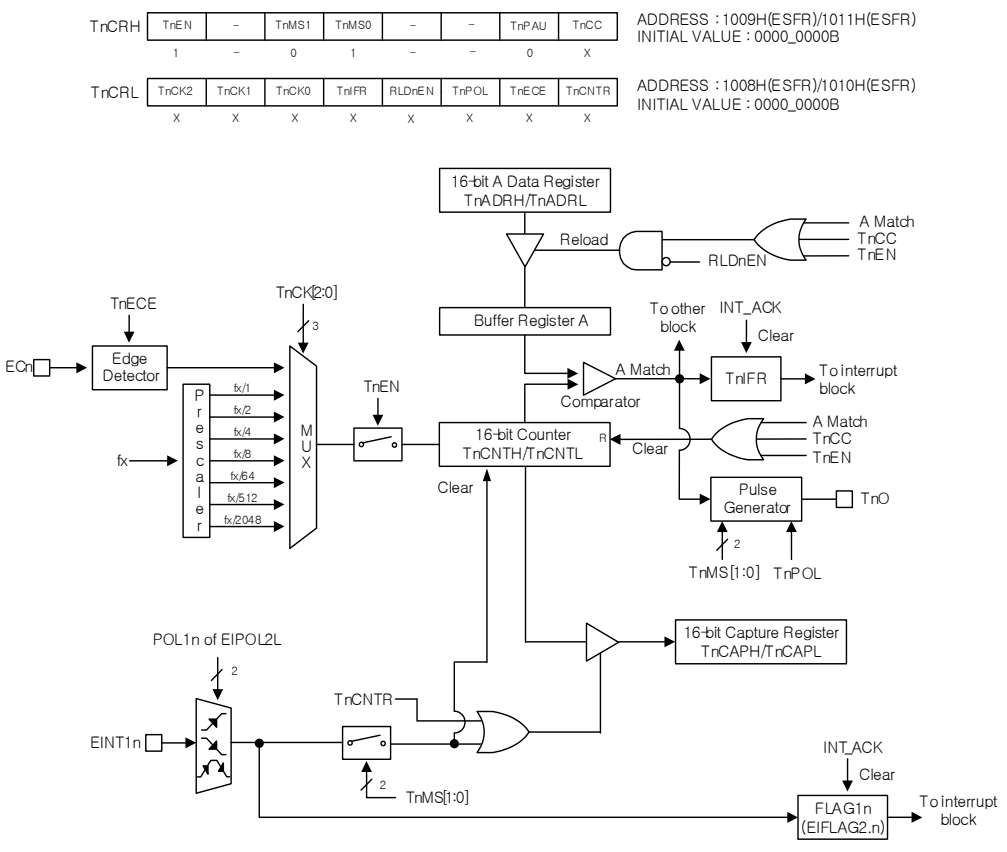


12.2 16-bit Capture Mode

16-bit timer 1/2 capture mode is set by configuring TnMS[1:0] as '01'. It uses an internal or external clock as a clock source. Basically, the 16-bit timer 1/2 capture mode has the same function as the 16-bit timer/counter mode, and the interrupt occurs when TnCNTH/TnCNTL is equal to TnADRH/TnADRL. The TnCNTH, TnCNTL values are automatically cleared by match signal. It can be cleared by software (TnCC) too.

A timer interrupt in capture mode is extremely useful when the pulse width of captured signal is wider than the maximum period of timer. Capture result is loaded into TnCAPH/L. According to EIPOL2L registers settings, the external interrupt EINT11/EINT12 function is selected. EINT11/EINT12 pin must be set as an input port.

Figure 40. 16-bit Capture Mode of TIMER 1/2 (where n=1, and 2, m=5, and 6)



NOTE:

1. RLDnEN bit must be '0' when the TnEN bit is changed from '0' to '1'.

Figure 41. Input Capture Mode Operation of TIMER 1/2

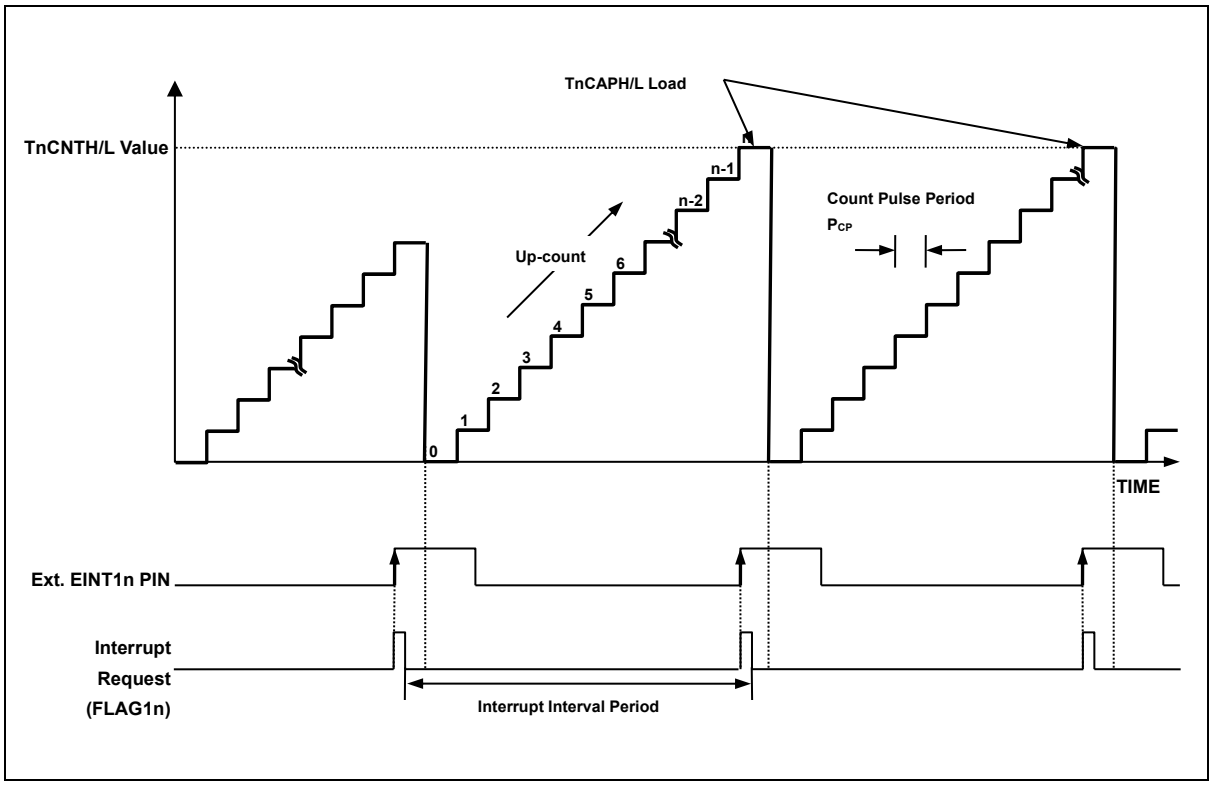
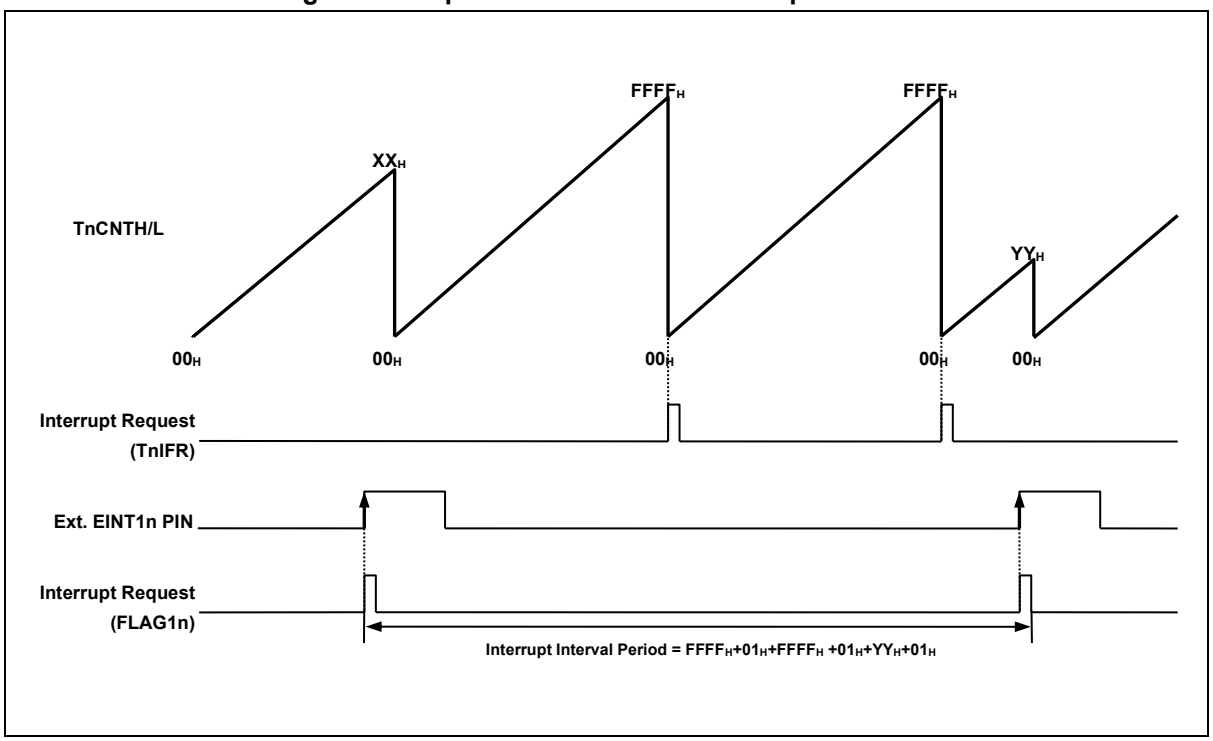


Figure 42. Express Timer Overflow in Capture Mode

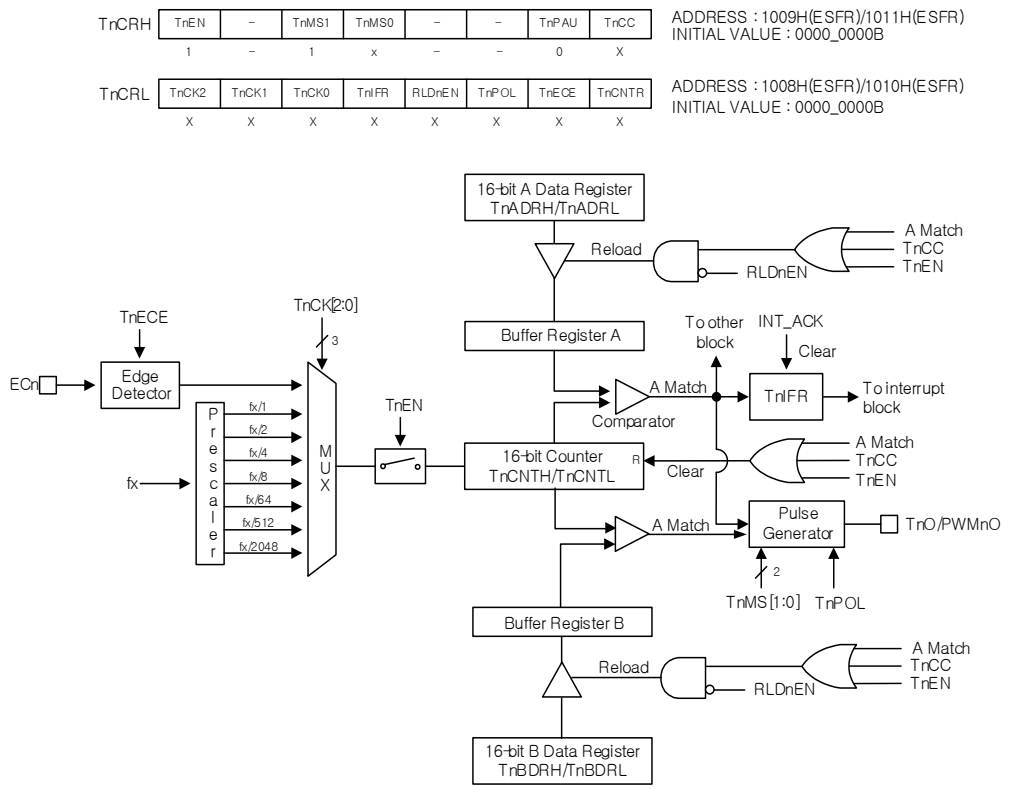


12.3 16-bit PPG Mode

TIMER 1/2 has a PPG (Programmable Pulse Generation) function. In PPG mode, TnO/PWMnO pin outputs up to 16-bit resolution PWM output.

For this function, TnO/PWMnO pin must be configured as a PWM output by setting P0FSRL6 to '1' (T1), P1FSRL6 to '0' (T2). Period of the PWM output is determined by TnADRH/TnADRL, and duty of the PWM output is determined by TnBDRH/TnBDRL.

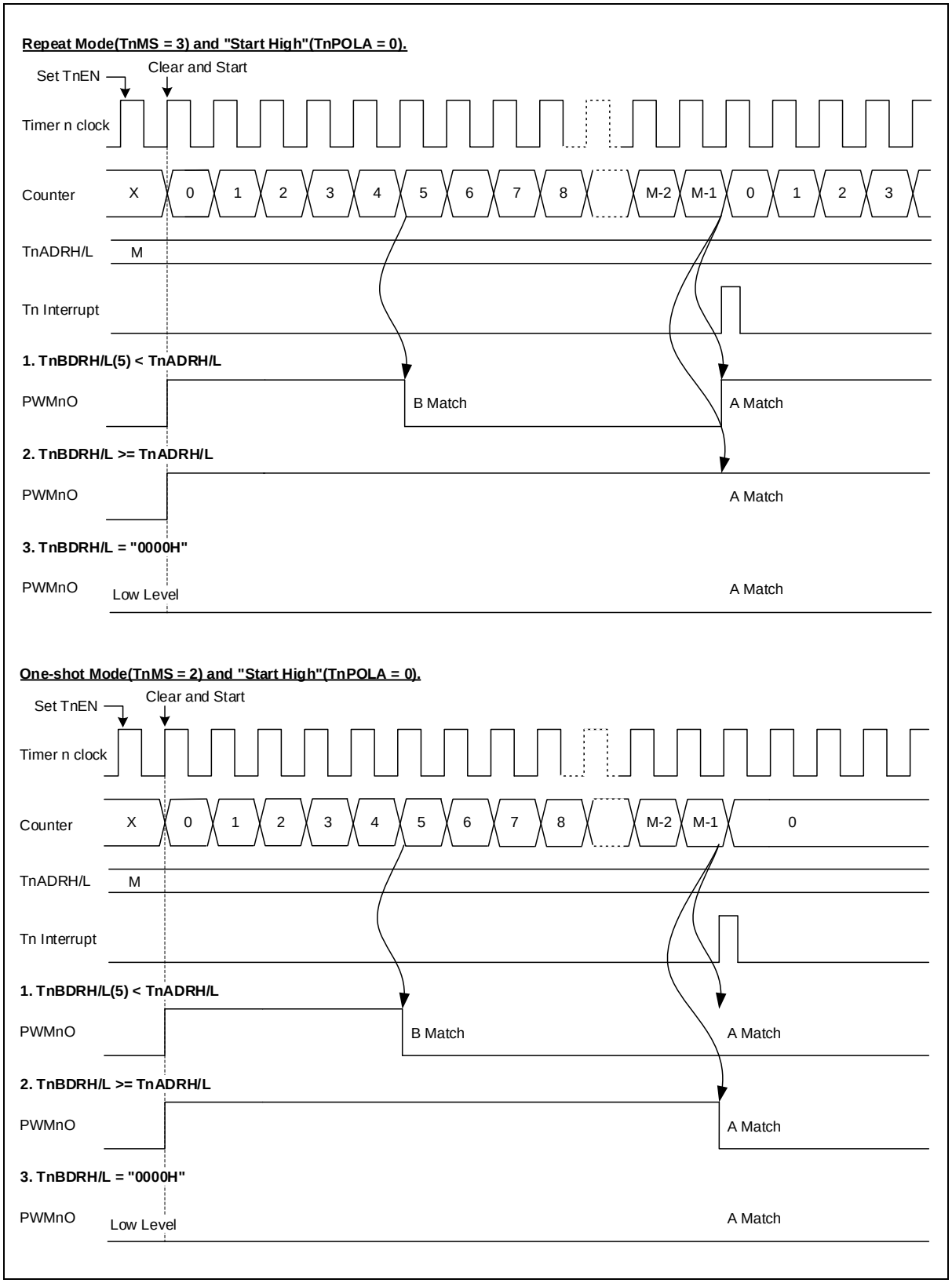
Figure 43. 16-bit PPG Mode of TIMER 1/2



NOTES:

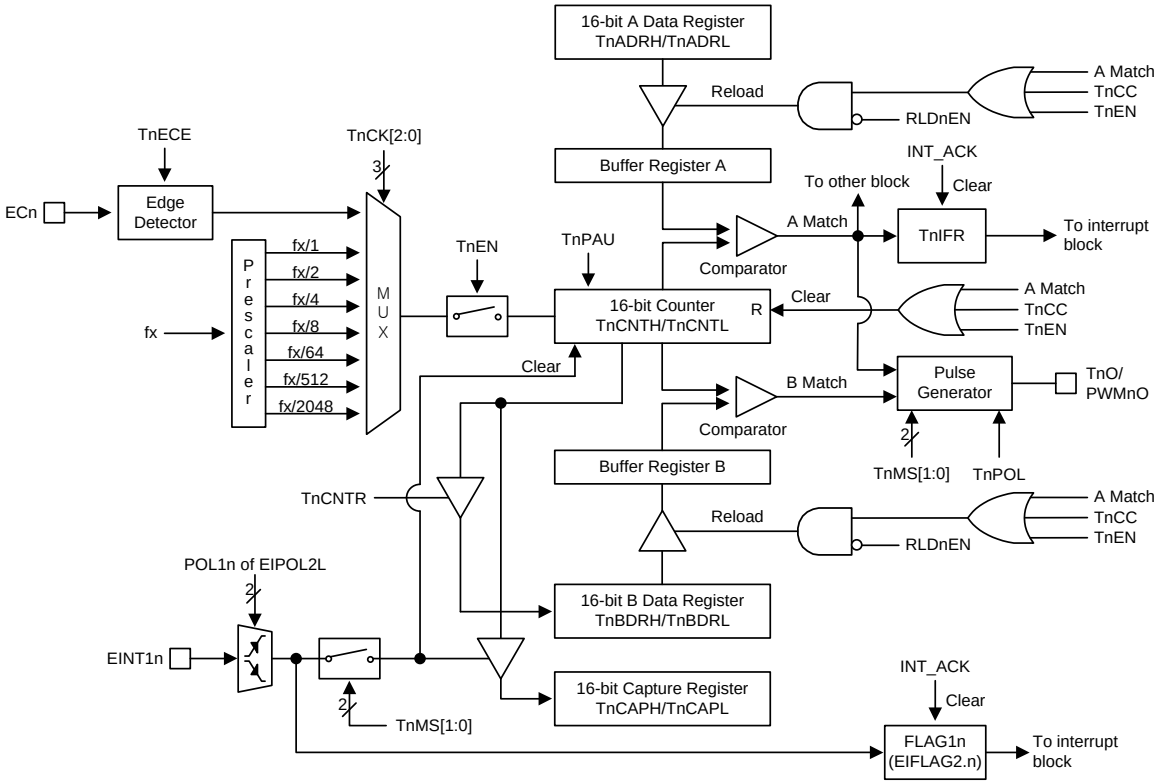
1. TnEN is automatically cleared to logic '0' after one pulse is generated at a PPG one-shot mode.
2. RLDnEN bit must be '0' when the TnEN bit is changed from '0' to '1'.

Figure 44. 16-bit PPG Mode Timing Chart of TIMER 1/2



12.4 Block Diagram

Figure 45. 16-bit Timer n in Block Diagram (where n = 1 and 2)



NOTES:

1. TnEN is automatically cleared to logic ‘0’ after one pulse is generated at a PPG one-shot mode.
2. RLDnEN bit must be ‘0’ when the TnEN bit is changed from ‘0’ to ‘1’.

12.5 Register Map

Table 20. TIMER n Register Map (where n = 1 and 2)

Name	Address	Direction	Default	Description
TnCRH	1009H/1011H(XSFR)	R/W	00H	Timer n Control High Register
TnCRL	1008H/1010H(XSFR)	R/W	00H	Timer n Control Low Register
TnADRH	100BH/1013H(XSFR)	R/W	FFH	Timer n A Data High Register
TnADRL	100AH/1012H(XSFR)	R/W	FFH	Timer n A Data Low Register
TnBDRH	100DH/1015H(XSFR)	R/W	FFH	Timer n B Data High Register
TnBDRL	100CH/1014H(XSFR)	R/W	FFH	Timer n B Data Low Register
TnCAPH	100FH/1017H(XSFR)	R	00H	Timer n Capture Data High Register
TnCAPL	100EH/1016H(XSFR)	R	00H	Timer n Capture Data Low Register

12.6 Timer/Counter 1/2 Register Description

TnADRH (Timer n A data High Register): 100BH/1013H (XSFR), where n = 1, and 2

7	6	5	4	3	2	1	0
TnADRH7	TnADRH6	TnADRH5	TnADRH4	TnADRH3	TnADRH2	TnADRH1	TnADRH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

TnADRH[7:0] Tn A Data High Byte

TnADRL (Timer n A Data Low Register): 100AH/1012H (XSFR), where n = 1, and 2

7	6	5	4	3	2	1	0
TnADRL7	TnADRL6	TnADRL5	TnADRL4	TnADRL3	TnADRL2	TnADRL1	TnADRL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

TnADRL[7:0] Tn A Data Low Byte

NOTE:

1. Do not write “0000H” in the TnADRH/TnADRL register when PPG mode

TnBDRH (Timer n B Data High Register): 100DH/1015H (XSFR), where n = 1, and 2

7	6	5	4	3	2	1	0
TnBDRH7	TnBDRH6	TnBDRH5	TnBDRH4	TnBDRH3	TnBDRH2	TnBDRH1	TnBDRH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

TnBDRH[7:0] Tn B Data High Byte

TnBDRL (Timer n B Data Low Register): 100CH/1014H (XSFR), where n = 1, and 2

7	6	5	4	3	2	1	0
TnBDRL7	TnBDRL6	TnBDRL5	TnBDRL4	TnBDRL3	TnBDRL2	TnBDRL1	TnBDRL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

TnBDRL[7:0] Tn B Data Low Byte

TnCAPH (Timer n Capture Data High Register): 100FH/1017H (XSFR), where n = 1, and 2

7	6	5	4	3	2	1	0
TnCAPH7	TnCAPH6	TnCAPH5	TnCAPH4	TnCAPH3	TnCAPH2	TnCAPH1	TnCAPH0
R	R	R	R	R	R	R	R

Initial value: 00H

TnCAPH[7:0] Tn Capture Data High Byte

TnCAPL (Timer n Capture Data Low Register): 100EH/1016H (XSFR), where n = 1, and 2

7	6	5	4	3	2	1	0
TnCAPL7	TnCAPL6	TnCAPL5	TnCAPL4	TnCAPL3	TnCAPL2	TnCAPL1	TnCAPL0
R	R	R	R	R	R	R	R

Initial value: 00H

TnCAPL[7:0] Tn Capture Data Low Byte

TnCRH (Timer n Control High Register): 1009H/1011H (XSFR), where n = 1, and 2

7	6	5	4	3	2	1	0
TnEN	–	TnMS1	TnMS0	–	–	TnPAU	TnCC
R/W	–	R/W	R/W	–	–	R/W	R/W

Initial value: 00H

TnEN Control Timer n
- 0 Timer n disable
- 1 Timer n enable (counter clear and start)

TnMS[1:0] Control Timer n Operation Mode

TnMS1	TnMS0	Description
0	0	Timer/counter mode (TnO: toggle at A match)
0	1	Capture mode (The A match interrupt can occur)
1	0	PPG one-shot mode (PWMnO)
1	1	PPG repeat mode (PWMnO)

TnPAU Timer n Counter Temporary Pause Control
- 0 Continue counting
- 1 Temporary pause

TnCC Clear Timer n Counter
- 0 No effect
- 1 Clear the Timer n counter (when write, automatically cleared “0” after being cleared counter)

TnCRL (Timer n Control Low Register): 1008H/1010H (XSFR), Where n = 1, and 2

7	6	5	4	3	2	1	0
TnCK2	TnCK1	TnCK0	TnIFR	RLDnEN	TnPOL	TnECE	TnCNTR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

TnCK[2:0] Select Timer n clock source. fx is main system clock frequency

TnCK2	TnCK1	TnCK0	Description
0	0	0	fx/2048
0	0	1	fx/512
0	1	0	fx/64
0	1	1	fx/8
1	0	0	fx/4
1	0	1	fx/2
1	1	0	fx/1
1	1	1	External clock (ECn)

TnIFR When Tn Interrupt occurs, this bit becomes ‘1’. For clearing bit, write ‘0’ to this bit or auto clear by INT_ACK signal.

0 Tn Interrupt no generation

1 Tn Interrupt generation

RLDnEN Control Timer n Reload Signal

0 Enable Timer n reload signal

1 Disable Timer n reload signal

TnPOL TnO/PWMnO Polarity Selection

0 Start High (TnO/PWMnO is low level at disable)

1 Start Low (TnO/PWMnO is high level at disable)

TnECE Timer n External Clock Edge Selection

0 External clock falling edge

1 External clock rising edge

TnCNTR Timer n Counter Read Control

0 No effect

1 Load the counter value to the B data register (when write, automatically cleared ‘0’ after being loaded)

13 12-bit A/D Converter

Analog-to-digital (A/D) converter allows conversion of an analog input signal to a corresponding 12-bit digital output. The A/D module has 8 analog inputs, and the output of the multiplexer becomes the input into the converter, which generates a result via successive approximation.

The A/D module incorporates four registers as listed in the following. Each register can be selected for the corresponding channel by setting ADSEL[3:0]. When conversion is completed, two registers ADCDRH and ADCDRL contain the results of the conversion, the conversion status bit AFLAG is set to '1', and A/D interrupt is set. During the A/D conversion, AFLAG bit is read as '0'.

- A/D converter control high register (ADCCRH)
- A/D converter control low register (ADCCRL)
- A/D converter data high register (ADCDRH)
- A/D converter data low register (ADCDRL)

13.1 Conversion Timing

The A/D conversion process requires 2 steps (2 clock edges) to convert each bit and 5 clocks to set up A/D conversion. Therefore, a total of 29 clocks are required to complete a 12-bit conversion: When fx/16 is selected for conversion clock with a 16 MHz fx clock frequency, one clock cycle is 1 μs. Each bit conversion requires two clocks, the conversion rate is calculated as follows:

2 clocks/bit × 12 bits + set-up time = 29 clocks,

29 clocks × 1μs = 29 μs at 1 MHz (16 MHz/16)

The conversion time of ADC can be set up to 9 [us] at VDD = 2.7 V to 3.6 V. If the system clock is 16 MHz and ADC clock is set to fx/5, the conversion clock of ADC is 3.2 MHz and conversion time is about 9 μs.

13.2 Block Diagram

Figure 46. 12-bit ADC Block Diagram

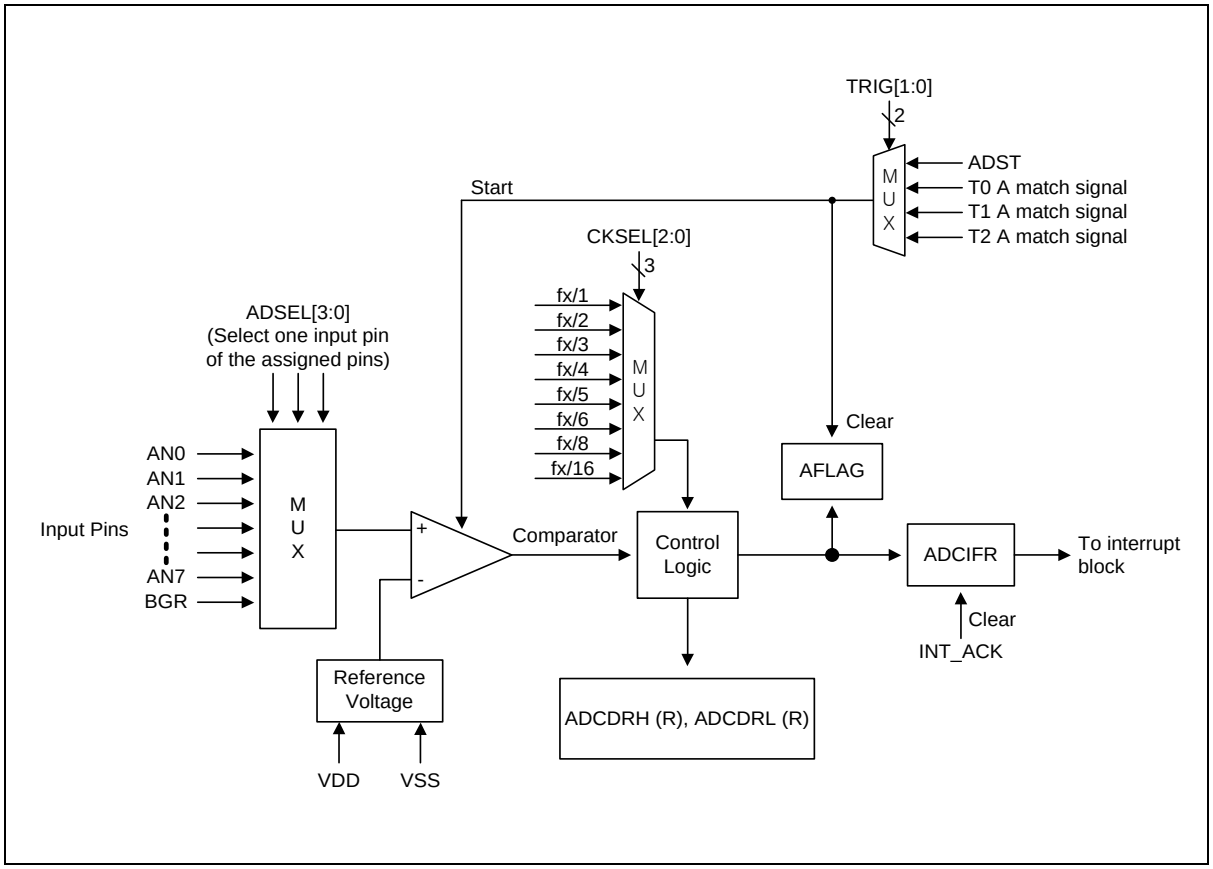
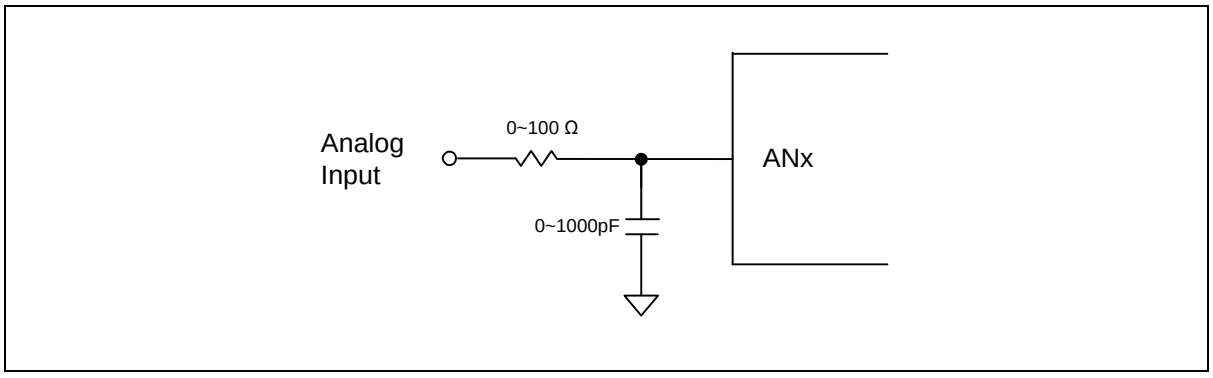


Figure 47. AD Analog Input Pin with Capacitor



13.3 ADC Operation

In this section, ADC operation is described Figure 48. As shown in Figure 48, ADC conversion starts after configuring ADC Control High/Low registers. By checking AFLAG, it is defined whether the conversion is completed or not. If AFLG is '1', the conversion is completed, and ADC Data high/low registers are read to finish ADC operation.

Figure 48. ADC Operation Flow

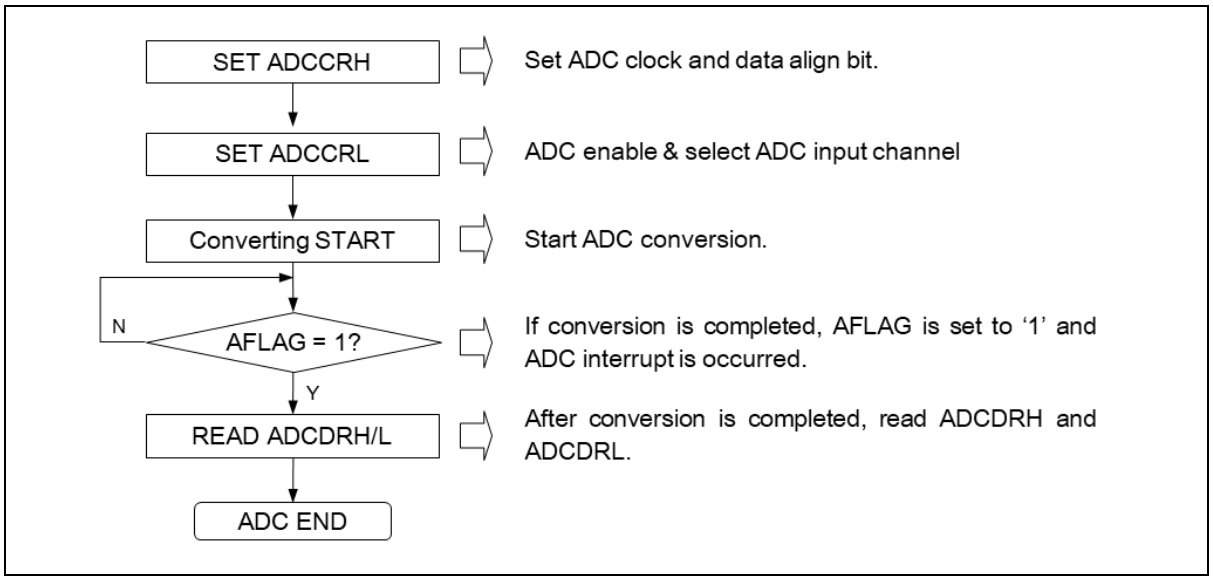
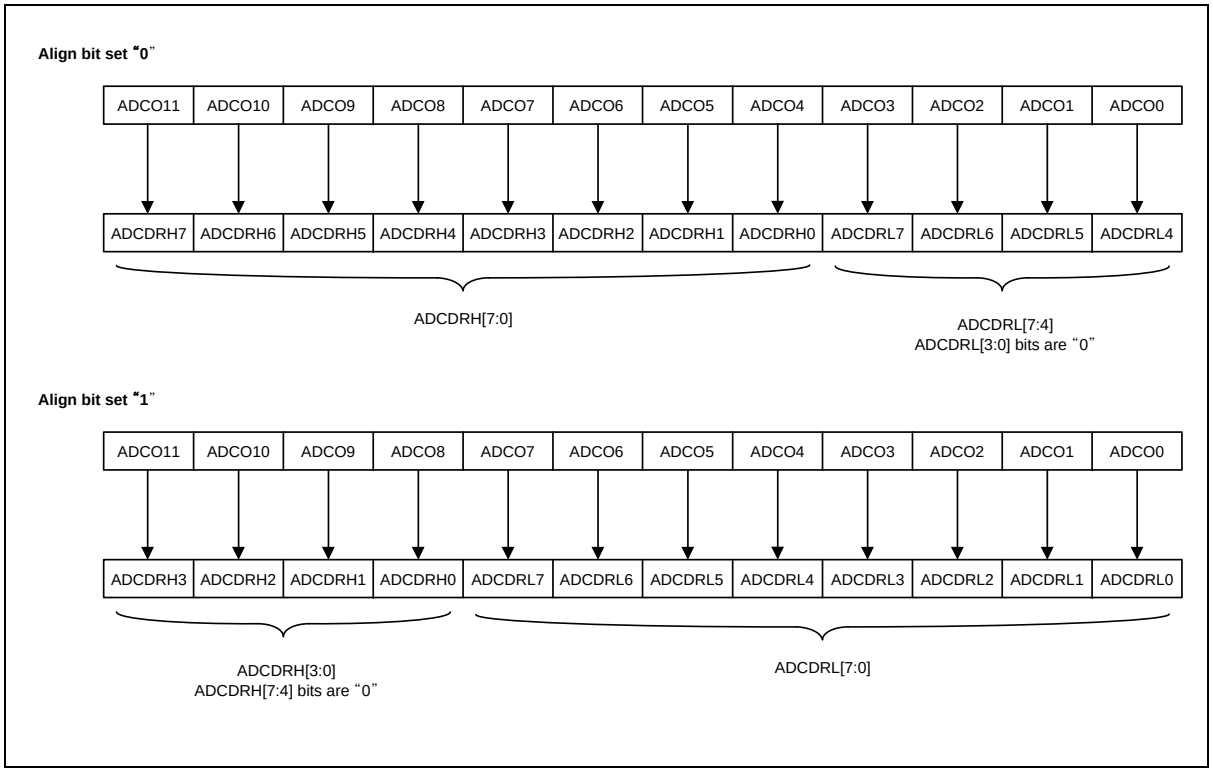


Figure 49. ADC Operation for Align Bit



13.4 Register Map

Table 21. 12-bit ADC Register Map

Name	Address	Direction	Default	Description
ADCCRH	93H	R/W	00H	A/D Converter Control High Register
ADCCRL	92H	R/W	00H	A/D Converter Control Low Register
ADCDRH	95H	R	xxH	A/D Converter Data High Register
ADCDRL	94H	R	xxH	A/D Converter Data Low Register

13.5 Register Description

ADCDRH (A/D Converter Data High Register): 95H

7	6	5	4	3	2	1	0
ADDM11	ADDM10	ADDM9	ADDM8	ADDM7 ADDL11	ADDM6 ADDL10	ADDM5 ADDL9	ADDM4 ADDL8
R	R	R	R	R	R	R	R

Initial value: xxH

ADDM[11:4]	MSB align, A/D Converter High Result (8-bit)
ADDL[11:8]	LSB align, A/D Converter High Result (4-bit)

ADCDRL (A/D Converter Data Low Register): 94H

7	6	5	4	3	2	1	0
ADDM3 ADDL7	ADDM2 ADDL6	ADDM1 ADDL5	ADDM0 ADDL4	ADDL3	ADDL2	ADDL1	ADDL0
R	R	R	R	R	R	R	R

Initial value: xxH

ADDM[3:0]	MSB align, A/D Converter Low Result (4-bit)
ADDL[7:0]	LSB align, A/D Converter Low Result (8-bit)

ADCCRH (A/D Converter Control High Register): 93H

7	6	5	4	3	2	1	0
ADCIFR	–	TRIG1	TRIG0	ALIGN	CKSEL2	CKSEL1	CKSEL0
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

ADCIFR — When ADC Interrupt occurs, this bit becomes ‘1’. For clearing bit, write ‘0’ to this bit or auto clear by INT_ACK signal. Writing “1” has no effect.

0	ADC Interrupt no generation
1	ADC Interrupt generation

TRIG[1:0] — A/D Converter Trigger Signal Selection

TRIG1	TRIG0	Description
0	0	ADST
0	1	Timer 0 A match signal
1	0	Timer 1 A match signal
1	1	Timer 2 A match signal

ALIGN — A/D Converter data align selection.

0	MSB align (ADCDRH[7:0], ADCDRL[7:4])
1	LSB align (ADCDRH[3:0], ADCDRL[7:0])

CKSEL[2:0] — A/D Converter Clock Selection

CKSEL2	CKSEL1	CKSEL0	Description
0	0	0	fx/1
0	0	1	fx/2
0	1	0	fx/3
0	1	1	fx/4
1	0	0	fx/5
1	0	1	fx/6
1	1	0	fx/8
1	1	1	fx/16

ADCCRL (A/D Converter Control Low Register): 92H

7	6	5	4	3	2	1	0
STBY	ADST	–	AFLAG	ADSEL3	ADSEL2	ADSEL1	ADSEL0
R/W	R/W	–	R	R/W	R/W	R/W	R/W

Initial value: 00H

STBY — Control Operation of A/D
(The ADC module is automatically disabled at stop mode)
0 ADC module disable
1 ADC module enable

ADST — Control Trigger Signal for Conversion Start.
0 No effect
1 Trigger signal generation for conversion start

AFLAG — A/D Converter Operation State (this bit is cleared to '0' when the STBY bit is set to '0' or when the CPU is at STOP mode)
0 During A/D Conversion
1 A/D Conversion finished

ADSEL[3:0] — A/D Converter input selection

ADSEL3	ADSEL2	ADSEL1	ADSEL0	Description
0	0	0	0	AN0
0	0	0	1	AN1
0	0	1	0	AN2
0	0	1	1	AN3
0	1	0	0	AN4
0	1	0	1	AN5
0	1	1	0	AN6
0	1	1	1	AN7
1	1	1	1	V_{BGR} (about 0.94 V)
Other values				Not available

14 I2C 0

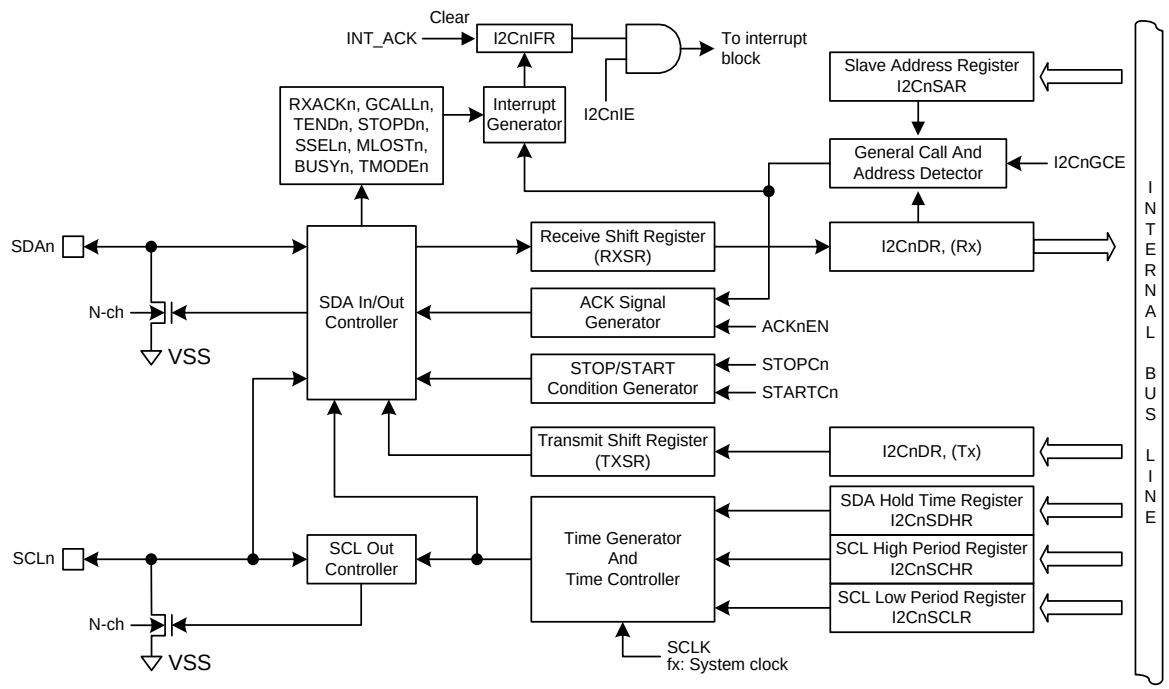
The I2C is one of the industrial standard serial communication protocols, and which uses two bus lines Serial Data Line (SDA) and Serial Clock Line (SCL) to exchange data. Because both SDAn and SCLn lines are open-drain output, each line needs a pull-up resistor.

The features are shown below.

- Compatible with I2C bus standard
- Multi-master operation
- Up to 400 kHz data transfer read speed
- 7-bit address
- Support two slave address
- Both master and slave operation
- Bus busy detection

14.1 Block Diagram

Figure 50. I2C Block Diagram (where n = 0)



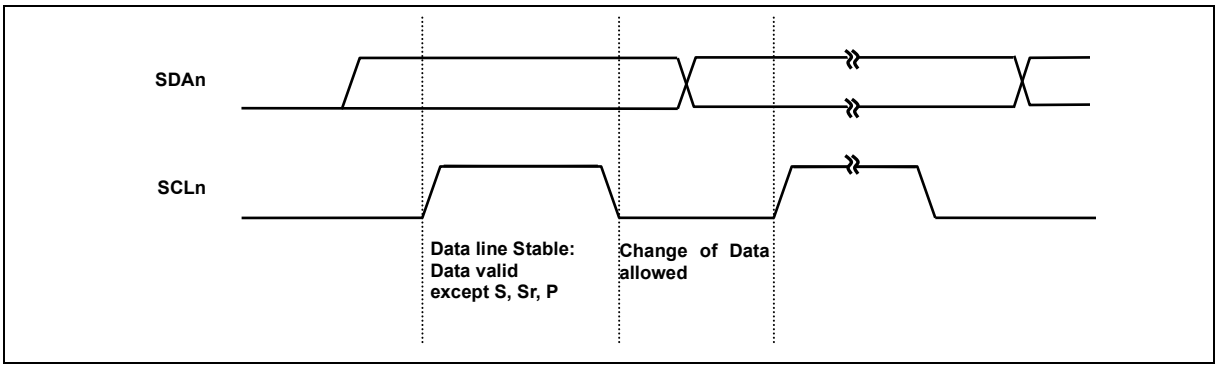
NOTE:

1. When the corresponding port of I2Cn is a sub-function for SCLn/SDAn pin, the SCLn/SDAn pins are automatically set to the N-channel open-drain outputs and the input latch is read in the case of reading the pins. The corresponding pull-up resistor is determined by the control register.

14.2 I2C Bit Transfer

The data on the SDAn line must be stable during HIGH period of the clock, SCLn. The HIGH or LOW state of the data line can only change when the clock signal in the SCLn line is LOW. The exceptions are START(S), repeated START(Sr) and STOP(P) condition where data line changes when clock line is high.

Figure 51. Bit Transfer in the I2C-Bus



14.3 Start/Repeated Start/Stop

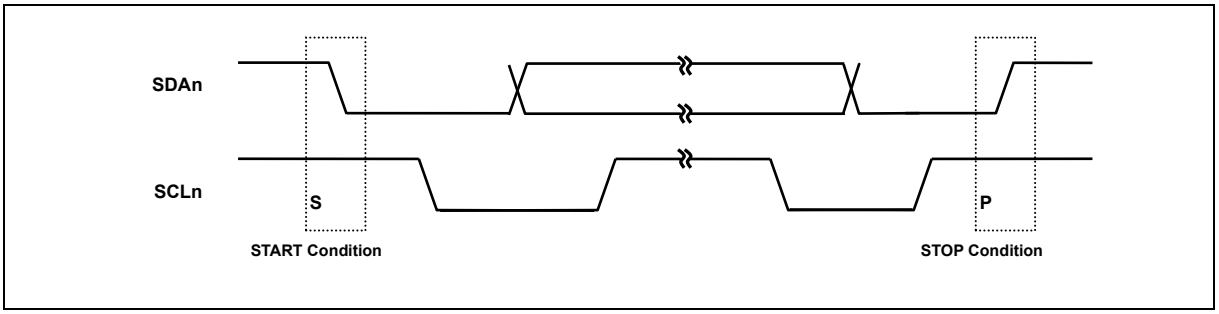
One master can issue a START (S) condition to recognize other devices connected to the SCLn, SDAn lines that it will use the bus. A STOP (P) condition is generated by the master to release the bus lines so that other devices can use it.

A high to low transition on the SDAn line while SCLn is high defines a START (S) condition.

A low to high transition on the SDAn line while SCLn is high defines a STOP (P) condition.

START and STOP conditions are always generated by the master. The bus is busy after START condition. The bus is free again after STOP condition, i.e., the bus is busy between START and STOP condition. If a repeated START condition (Sr) is generated instead of STOP condition, the bus stays busy. So, the START and repeated START conditions are functionally identical.

Figure 52. START and STOP Condition



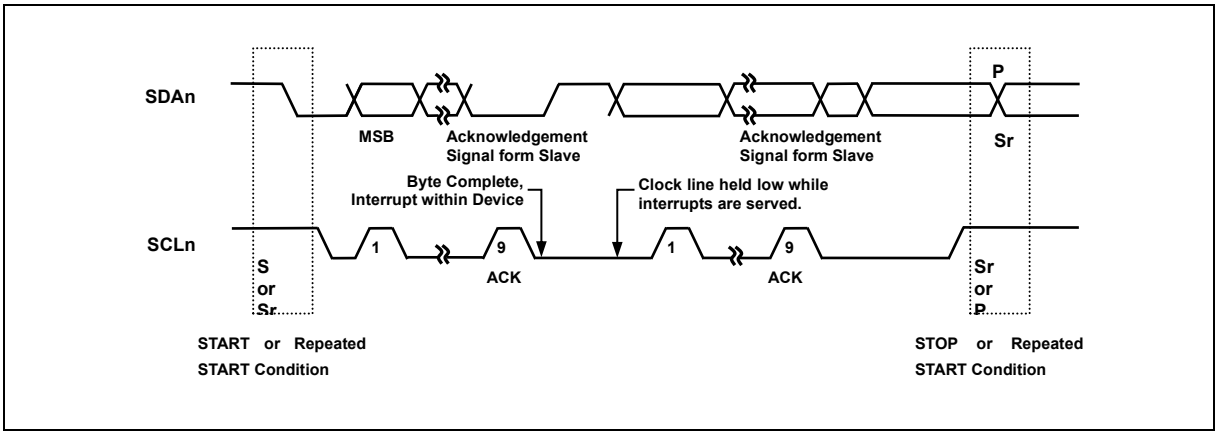
14.4 Data Transfer

Every byte put on the SDAn line must be 8-bits long. The number of bytes that can be transmitted per transfer is unlimited. Each byte has to be followed by an Acknowledge bit. Data is transferred with the most significant bit (MSB) first.

If a slave can't receive or transmit another complete byte of data until it has performed some other function, it can hold the clock line SCLn LOW to force the master into a wait state.

Data transfer then continues when the slave is ready for another byte of data and releases clock line.

Figure 53. Data Transfer on the I2C-Bus



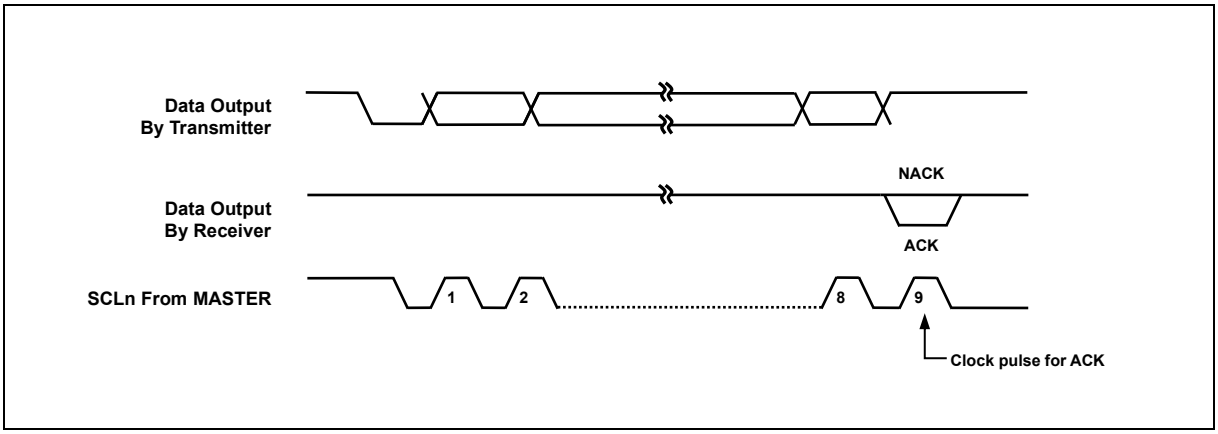
14.5 I2C Acknowledge

The acknowledge related clock pulse is generated by the master. The transmitter releases the SDAn line (HIGH) during the acknowledge clock pulse. The receiver must pull down the SDAn line during the Acknowledge clock pulse so that it remains stable LOW during the HIGH period of this clock pulse.

When a slave is addressed by a master (Address Packet), and if it is unable to receive or transmit because it's performing some real time function, the data line must be left HIGH by the slave. And, when a slave addressed by a master is unable to receive more data bits, the slave receiver must release the SDAn line (Data Packet). The master can then generate either a STOP condition to abort the transfer, or a repeated START condition to start a new transfer.

If a master receiver is involved in a transfer, it must signal the end of data to the slave transmitter by not generating an acknowledge on the last byte that was clocked out of the slave. The slave transmitter must release the data line to allow the master to generate a STOP or repeated START condition.

Figure 54. Acknowledge on the I2C-Bus



14.6 Synchronization and Arbitration

Clock synchronization is performed using the wired-AND connection of I2C interfaces to the SCLn line. This means that a HIGH to LOW transition on the SCLn line will cause the devices concerned to start counting off their LOW period and it will hold the SCLn line in that state until the clock HIGH state is reached. However, the LOW to HIGH transition of this clock may not change the state of the SCLn line if another clock is still within its LOW period. In this way, a synchronized SCLn clock is generated with its LOW period determined by the device with the longest clock LOW period, and its HIGH period determined by the one with the shortest clock HIGH period.

A master may start a transfer only if the bus is free. Two or more masters may generate a START condition. Arbitration takes place on the SDAn line, while the SCL nline is at the HIGH level, in such a way that the master which transmits a HIGH level, while another master is transmitting a LOW level will switch off its DATA output state because the level on the bus doesn't correspond to its own level. Arbitration continues for many bits until a winning master gets the ownership of I2C bus. Its first stage is comparison of the address bits.

Figure 55. Clock Synchronization during Arbitration Procedure

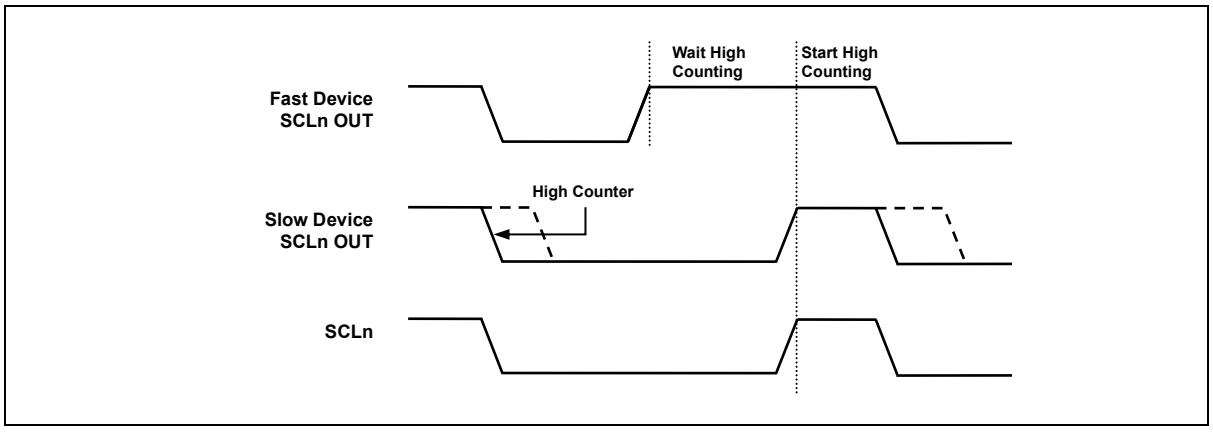
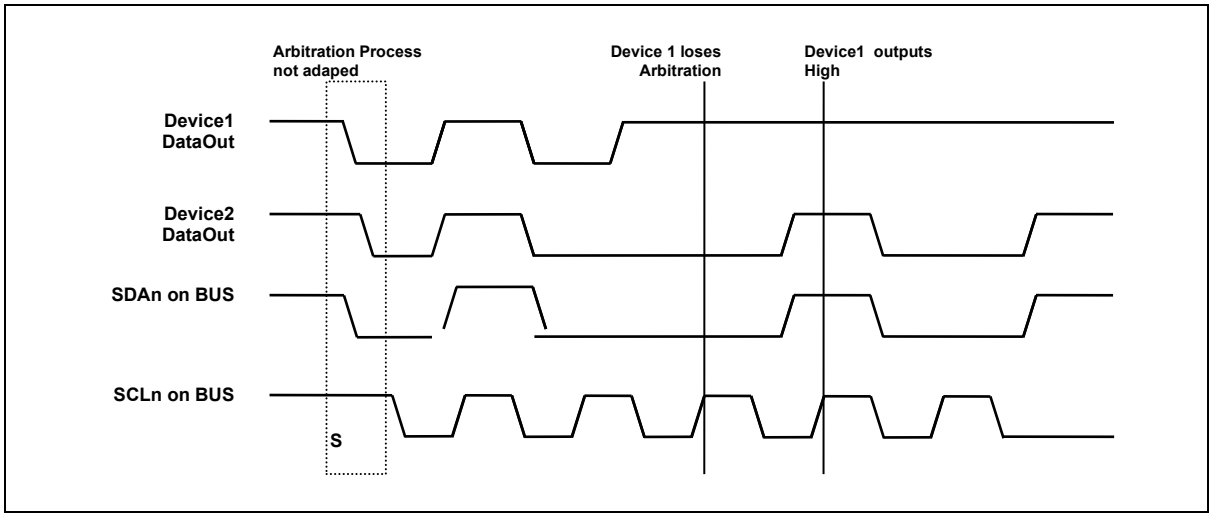


Figure 56. Arbitration Procedure of Two Masters.



14.7 Operation

The I2C operates in byte units and is based on interrupts. The interrupts are issued after all bus events except for a transmission of a START condition. Because the I2C is interrupt based, the application software is free to carry on other operations during an I2C byte transfer.

Please remember that when the I2C interrupt is generated, the I2CnIFR flag in I2CnFLAG register is set and it is cleared when all interrupt source bits in the I2CnSR register are cleared to “0b”. When the I2C interrupt occurs, the SCLn line is held LOW until clearing “0b” all interrupt source bits in I2CnSR register. When the I2CnIFR flag is set, the I2CnSR contains a value indicating the current state of the I2C bus. According to the value in I2CnSR, software can decide what to do next.

The I2C can operate in 4 modes by configuring master/slave, transmitter/receiver. The operating mode is configured by a winning master. A more detailed explanation follows below.

14.8 Master Transmitter

To operate I2C in master transmitter, follow the recommended steps below.

1. Enable I2C by setting I2CnEN bit in I2CnCR. This provides the main clock to the peripheral.
2. Load SLA+W into the I2CnDR where SLA is address of slave device and W is transfer direction from the viewpoint of the master. For master transmitter, W is ‘0’. Note that I2CnDR is used for both address and data.
3. Configure baud rate by writing desired value to both I2CnSCLR and I2CnSCHR for the Low and High period of SCLn line.
4. Configure the I2CnSDHR to decide when SDAn changes value from falling edge of SCLn. If SDAn should change in the middle of SCLn LOW period, load half the value of I2CnSCLR to the I2CnSDHR.
5. Set the STARTC bit in I2CnCR. This transmits a START condition. And configure how to handle interrupt and ACK signal. When the STARTCn bit is set, 8-bit data in I2CnDR is transmitted out according to the baud-rate.
6. This is ACK signal processing stage for address packet transmitted by master. When a 7-bit address and 1-bit transfer direction is transmitted to target slave device, the master can know whether the slave acknowledged or not in the 9th high period of SCLn. If the master gains bus mastership, I2C generates GCALLn interrupt regardless of the reception of ACK from the slave device. When I2C loses bus mastership during arbitration process, the MLOSTn bit in I2CnSR is set, and I2C waits in idle state or can be operate as an addressed slave.

   To operate as a slave when the MLOSTn bit in I2CnSR is set, the ACKnEN bit in I2CnCR must be set and the received 7-bit address must equal to the SLAn bits in I2CnSAR. In this case I2C operates as a slave transmitter or a slave receiver (go to appropriate section). In this stage, I2C holds the SCLn LOW. This is because to decide whether I2C continues serial transfer or stops communication. The following steps continue assuming that I2C does not lose mastership during first data transfer.

I2C (Master) can choose one of the following cases regardless of the reception of ACK signal from slave.

A. Master receives ACK signal from slave, so continues data transfer because slave can receive more data from master. In this case, load data to transmit to I2CnDR.

B. Master stops data transfer even if it receives ACK signal from slave. In this case, set the STOPCn bit in I2CnCR.

C. Master transmits repeated START condition without checking ACK signal. In this case, load SLA+R/W into the I2CnDR and set STARTCn bit in I2CnCR.

After doing one of the actions above, clear to “0b” all interrupt source bits in I2CnSR to release SCLn line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CnDR and if transfer direction bit is ‘1’ go to master receiver section.

7. 1-Byte of data is being transmitted. During data transfer, bus arbitration continues.

8. This is ACK signal processing stage for data packets transmitted by master. I2C holds the SCLn LOW. When I2C loses bus mastership while transmitting data arbitrating other masters, the MLOSTn bit in I2CnSR is set. If then, I2C waits in idle state. When the data in I2CnDR is transmitted completely, I2C generates TENDn interrupt.

   I2C can choose one of the following cases regardless of the reception of ACK signal from slave.

   A. Master receives ACK signal from slave, so continues data transfer because slave can receive more data from master. In this case, load data to transmit to I2CnDR.

   B. Master stops data transfer even if it receives ACK signal from slave. In this case, set the STOPCn bit in I2CnCR.

   C. Master transmits repeated START condition without checking ACK signal. In this case, load SLA+R/W into the I2CnDR and set the STARTCn bit in I2CnCR.

   After doing one of the actions above, clear to “0b” all interrupt source bits in I2CnSR to release SCLn line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CnDR, and if transfer direction bit is ‘1’ go to master receiver section.

9. This is the final step for master transmitter function of I2C, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CnSR, write “0” to I2CnSR. After this, I2C enters idle state.

14.9 Master Receiver

To operate I2C in master receiver, follow the recommended steps below.

1. Enable I2C by setting I2CnEN bit in I2CnCR. This provides the main clock to the peripheral.
2. Load SLA+R into the I2CnDR where SLAn is address of slave device and R is transfer direction from the viewpoint of the master. For master receiver, R is '1'. Note that I2CnDR is used for both address and data.
3. Configure baud rate by writing desired value to both I2CnSCLR and I2CnSCHR for the Low and High period of SCLn line.
4. Configure the I2CnSDHR to decide when SDAn changes value from falling edge of SCLn. If SDAn should change in the middle of SCLn LOW period, load half the value of I2CnSCLR to the I2CnSDHR.
5. Set the STARTCn bit in I2CnCR. This transmits a START condition. And configure how to handle interrupt and ACK signal. When the STARTCn bit is set, 8-bit data in I2CnDR is transmitted out according to the baud-rate.
6. This is ACK signal processing stage for address packet transmitted by master. When a 7-bit address and 1-bit transfer direction is transmitted to target slave device, the master can know whether the slave acknowledged or not in the 9th high period of SCLn. If the master gains bus mastership, I2C generates GCALLn interrupt regardless of the reception of ACK from the slave device. When I2C loses bus mastership during arbitration process, the MLOSTn bit in I2CnSR is set, and I2C waits in idle state or can be operate as an addressed slave.

   To operate as a slave when the MLOST bit in I2CnSR is set, the ACKnEN bit in I2CnCR must be set and the received 7-bit address must equal to the SLAn bits in I2CnSAR. In this case I2C operates as a slave transmitter or a slave receiver (go to appropriate section). In this stage, I2C holds the SCLn LOW. This is because to decide whether I2C continues serial transfer or stops communication. The following steps continue if the I2C does not lose mastership during first data transfer.

   I2C (Master) can choose one of the following cases according to the reception of ACK signal from slave.

   A. Master receives ACK signal from slave, so continues data transfer because slave can prepare and transmit more data to master. Configure ACKnEN bit in I2CnCR to decide whether I2C Acknowledges the next data to be received or not.

   B. Master stops data transfer because it receives no ACK signal from slave. In this case, set the STOPCn bit in I2CnCR.

   C. Master transmits repeated START condition due to no ACK signal from slave. In this case, load SLA+R/W into the I2CnDR and set STARTCn bit in I2CnCR.

After doing one of the actions above, clear to “0b” all interrupt source bits in I2CnSR to release SCLn line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CnDR and if transfer direction bit is ‘0’ go to master transmitter section.

7. 1-Byte of data is being received.

8. This is ACK signal processing stage for data packets transmitted by slave. I2C holds the SCLn LOW. When 1-Byte of data is received completely, I2C generates TENDn interrupt.

   I2C can choose one of the following cases according to the RXACKn flag in I2CnSR.

   A. Master continues receiving data from slave. To do this, set ACKnEN bit in I2CnCR to Acknowledge the next data to be received.

   B. Master wants to terminate data transfer when it receives next data by not generating ACK signal. This can be done by clearing ACKnEN bit in I2CnCR.

   C. Because no ACK signal is detected, master terminates data transfer. In this case, set the STOPCn bit in the I2CnCR register.

   D. No ACK signal is detected, and master transmits repeated START condition. In this case, load SLA+R/W into the I2CnDR and set the STARTCn bit in the I2CnCR register.

   After doing one of the actions above, clear to ‘0’ all interrupt source bits in the I2CnSR register to release SCLn line. In case of 1) and 2), move to step 7. In case of 3), move to step 9 to handle STOP interrupt. In case of 4), move to step 6 after transmitting the data in the I2CnDR register, and if transfer direction bit is ‘0’ go to master transmitter section.

9. This is the final step for master receiver function of I2C, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear the I2CnSR register, write ‘0’ value to the I2CnSR register. After this, I2C enters idle state.

14.10 Slave Transmitter

To operate I2C in slave transmitter, follow the recommended steps below.

1. If the main operating clock (SCLK) of the system is slower than that of SCLn, load value 0x00 into I2CnSDHR to make SDAn change within one system clock period from the falling edge of SCLn. Note that the hold time of SDAn is calculated by SDAH x period of SCLK where SDAH is multiple of number of SCLK coming from I2CnSDHR. When the hold time of SDAn is longer than the period of SCLK, I2C (slave) cannot transmit serial data properly.
2. Enable I2C by setting I2CnIE bit and I2CnEN bit in I2CnCR. This provides the main clock to the peripheral.
3. When a START condition is detected, I2C receives one byte of data and compares it with SLAn bits in I2CnSAR. If the GCALLnEN bit in I2CnSAR is enabled, I2C compares the received data with value 0x00, the general call address.
4. If the received address does not equal to SLAn bits in I2CnSAR, I2C enters idle state i.e., waits for another START condition. Else if the address is equal to the SLAn bits and the ACKnEN bit is enabled, I2C generates SSELn interrupt and the SCLn line is held LOW. Note that even if the address equals to SLAn bits, when the ACKnEN bit is disabled, I2C enters idle state. When SSELn interrupt occurs, load transmit data to I2CnDR and clear to “0b” all interrupt source bits in I2CnSR to release SCLn line.
5. 1-Byte of data is being transmitted.
6. In this step, I2C generates TENDn interrupt and holds the SCLn line LOW regardless of the reception of ACK signal from master. Slave can select one of the following cases. I2C (Master) can choose one of the following cases according to the reception of ACK signal from slave.

   A. No ACK signal is detected and I2C waits for STOP or repeated START condition.

   B. ACK signal from master is detected. Load data to transmit into I2CnDR.

   After doing one of the actions above, clear to “0b” all interrupt source bits in I2CnSR to release SCLn line. In case of 1) move to step 7 to terminate communication. In the case of 2) move to step 5. In either case, a repeated START condition can be detected. For that case, move to step 4.
7. This is the final step for slave transmitter function of I2C, handling STOP interrupt. The STOPCn bit indicates that data transfer between master and slave is over. To clear I2CnSR, write “0” to I2CnSR. After this, I2C enters idle state.

14.11 Slave Receiver

To operate I2C in slave receiver, follow the recommended steps below.

1. If the main operating clock (SCLK) of the system is slower than that of SCLn, load value 0x00 into I2CnSDHR to make SDAn change within one system clock period from the falling edge of SCLn. Note that the hold time of SDAn is calculated by SDAH x period of SCLK where SDAH is multiple of number of SCLK coming from I2CnSDHR. When the hold time of SDAn is longer than the period of SCLK, I2C (slave) cannot transmit serial data properly.
2. Enable I2C by setting I2CnIE bit in I2CnCR. This provides the main clock to the peripheral.
3. When a START condition is detected, I2C receives one byte of data and compares it with SLAn bits in I2CnSAR. If the GCALLnEN bit in I2CnSAR is enabled, I2C compares the received data with value 0x00, the general call address.
4. If the received address does not equal to SLAn bits in I2CnSAR, I2C enters idle state i.e., waits for another START condition. Else if the address is equal to the SLAn bits and the ACKnEN bit is enabled, I2C generates SSELn interrupt and the SCLn line is held LOW. Note that even if the address equals to SLAn bits, when the ACKnEN bit is disabled, I2C enters idle state. When SSELn interrupt occurs and I2C is ready to receive data, clear to “0b” all interrupt source bits in I2CnSR to release SCLn line.
5. 1-Byte of data is being received.
6. In this step, I2C generates TENDn interrupt and holds the SCLn line LOW regardless of the reception of ACK signal from master. Slave can select one of the following cases. I2C (Master) can choose one of the following cases according to the reception of ACK signal from slave.

   A. No ACK signal is detected (ACKnEN=0) and I2C waits STOP or repeated START condition.

   B. ACK signal is detected (ACKnEN=1) and I2C can continue to receive data from master.

   After doing one of the actions above, clear to ‘0’ all interrupt source bits in I2CnSR to release SCLn line. In case of 1) move to step 7 to terminate communication. In the case of 2) move to step 5. In either case, a repeated START condition can be detected. For that case, move to step 4.
7. This is the final step for slave receiver function of I2C, handling STOP interrupt. The STOPCn bit indicates that data transfer between master and slave is over. To clear I2CnSR, write “0” to I2CnSR. After this, I2C enters idle state.

14.12 Register Map

Table 22. I2C Register Map

Name	Address	Direction	Default	Description
I2CnCR	EAH	R/W	00H	I2Cn Control Register
I2CnSR	EBH	R/W	00H	I2Cn Status Register
I2CnSAR0	F6H	R/W	00H	I2Cn Slave Address 0 Register
I2CnSAR1	F7H	R/W	00H	I2Cn Slave Address 1 Register
I2CnDR	ECH	R/W	00H	I2Cn Data Register
I2CnSDHR	EDH	R/W	01H	I2Cn SDA Hold Time Register
I2CnSCLR	EEH	R/W	3FH	I2Cn SCL Low Period Register
I2CnSCHR	EFH	R/W	3FH	I2Cn SCL High Period Register

14.13 Register Description

I2CnDR (I2Cn Data Register): ECH, n = 0

7	6	5	4	3	2	1	0
I2CnDR7	I2CnDR6	I2CnDR5	I2CnDR4	I2CnDR3	I2CnDR2	I2CnDR1	I2CnDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

I2CnDR[7:0]	The I2CnDR transmit buffer and receive buffer share the same I/O address with this DATA register. The transmit data buffer is the destination for data written to the I2CnDR register. Reading the I2CnDR register returns the contents of the receive buffer.

I2CnSDHR (I2Cn SDA Hold Time Register): EDH, n = 0

7	6	5	4	3	2	1	0
I2CnSDHR7	I2CnSDHR6	I2CnSDHR5	I2CnSDHR4	I2CnSDHR3	I2CnSDHR2	I2CnSDHR1	I2CnSDHR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 01H

I2CnSDHR[7:0]	The register is used to control SDAn output timing from the falling edge of SCLn. Note that SDAn is changed after t_{SCLK} × (I2CnSDHR+2), in master mode, load half the value of I2CnSCLR to this register to make SDAn change in the middle of SCLn. In slave mode, configure this register regarding the frequency of SCLn from master. The SDAn is changed after t_{SCLK} × (I2CnSDHR+2) in master mode. So, to ensure operation in slave mode, the value. t_{SCLK} × (I2CnSDHR +2) must be smaller than the period of SCLn.

I2CnSCHR (I2Cn SCL High Period Register): EFH, n = 0

7	6	5	4	3	2	1	0
I2CnSCHR7	I2CnSCHR6	I2CnSCHR5	I2CnSCHR4	I2CnSCHR3	I2CnSCHR2	I2CnSCHR1	I2CnSCHR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 3FH

I2CnSCHR[7:0] This register defines the high period of SCLn in master mode. The base clock is SCLK, the system clock, and the period is calculated by the formula: tSCLK × (4 × I2CnSCHR + 2) where tSCLK is the period of SCLK.

So, the operating frequency of I2C master mode is calculated by the following equation.

$$f_{I2C} = \frac{1}{t_{SCLK} \times (4 \times (I2CnSCLR + I2CnSCHR) + 4)}$$

I2CnSCLR (I2Cn SCL Low Period Register): EEH, n = 0

7	6	5	4	3	2	1	0
I2CnSCLR7	I2CnSCLR6	I2CnSCLR5	I2CnSCLR4	I2CnSCLR3	I2CnSCLR2	I2CnSCLR1	I2CnSCLR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 3FH

I2CnSCLR[7:0] This register defines the low period of SCLn in master mode. The base clock is SCLK, the system clock, and the period is calculated by the formula: tSCLK × (4 × I2CnSCLR + 2) where tSCLK is the period of SCLK.

I2CnSAR0 (I2Cn Slave Address 0 Register): F6H, n = 0

7	6	5	4	3	2	1	0
I2CnSLA06	I2CnSLA05	I2CnSLA04	I2CnSLA03	I2CnSLA02	I2CnSLA01	I2CnSLA00	GCALLnEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

I2CnSLA0[6:0] These bits configure the slave address 0 in slave mode.

GCALLnEN This bit decides whether I2Cn allows general call address or not in I2Cn slave mode.

0 Ignore general call address

1 Allow general call address

I2CnSAR1 (I2Cn Slave Address 1 Register): F7H, n = 0

7	6	5	4	3	2	1	0
I2CnSLA16	I2CnSLA15	I2CnSLA14	I2CnSLA13	I2CnSLA12	I2CnSLA11	I2CnSLA10	GCALLnEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

I2CnSLA1[6:0] These bits configure the slave address 1 in slave mode.

GCALLnEN This bit decides whether I2Cn allows general call address or not in I2Cn slave mode.

0 Ignore general call address

1 Allow general call address

I2CnCR (I2Cn Control Register): EAH, n = 0

7	6	5	4	3	2	1	0
I2CnRST	I2CnEN	TXDLYENBn	I2CnIE	ACKnEN	IMASTERn	STOPCn	STARTCn
R/W	R/W	R/W	R/W	R/W	R	R/W	R/W

Initial value: 00H

I2CnRST Initialize Internal Registers of I2Cn.

0 No effect

1 Initialize I2Cn, auto cleared

I2CnEN Activate I2Cn Function Block by Supplying

0 I2Cn is disabled

1 I2Cn is enabled

TXDLYENBn I2CnSDHR register control bit

0 Enable I2CnSDHR Register

1 Disable I2CnSDHR Register

I2CnIE Interrupt Enable bit.

0 Interrupt of I2Cn is inhibited (use polling)

1 Enable interrupt for I2Cn

ACKnEN Controls ACK signal generation at ninth SCLn period.

0 No ACK signal is generated (SDAn=1)

1 ACK signal is generated (SDAn=0)

NOTES: ACK signal is output (SDAn=0) for the following 3 cases.

1. When received address packet equals to I2CnSLA bits in I2CnSAR.
2. When received address packet equals to value 0x00 with GCALLn enabled.
3. When I2Cn operates as a receiver (master or slave)

IMASTERn Represent operating mode of I2Cn

0 I2Cn is in slave mode

1 I2Cn is in master mode

STOPCn When I2Cn is master, STOP condition generation

0 No effect

1 STOP condition is to be generated

STARTCn When I2Cn is master, START condition generation

0 No effect

1 START condition is to be generated

NOTE:

1. Refer to the external interrupt flag register (EIFLAG1) for the I2Cn interrupt flag.

I2CnSR (I2Cn Status Register): EBH, n = 0

7	6	5	4	3	2	1	0
GCALLn	TENDn	STOPDn	SSELn	MLOSTn	BUSYn	TMODEn	RXACKn
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W

Initial value: 00H

GCALLn [NOTE] This bit has different meaning depending on whether I2C is master or slave. When I2C is a master, this bit represents whether it received AACK (address ACK) from slave
- 0 No AACK is received (Master mode)
- 1 AACK is received (Master mode)

When I2C is a slave, this bit is used to indicate general call
- 0 General call address is not detected (Slave mode)
- 1 General call address is detected (Slave mode)

TENDn [NOTE] This bit is set when 1-byte of data is transferred completely
- 0 1 byte of data is not completely transferred
- 1 1 byte of data is completely transferred

STOPDn [NOTE] This bit is set when a STOP condition is detected
- 0 No STOP condition is detected
- 1 A STOP condition is detected

SSELn [NOTE] This bit is set when I2C is addressed by other master
- 0 I2C is not selected as a slave
- 1 I2C is addressed by other master and acts as a slave

MLOSTn [NOTE] This bit represents the result of bus arbitration in master mode
- 0 I2C maintains bus mastership
- 1 I2C has lost bus mastership during arbitration process

BUSYn This bit reflects bus status
- 0 I2C bus is idle, so a master can issue a START condition
- 1 I2C bus is busy

TMODEn This bit is used to indicate whether I2C is transmitter or receiver
- 0 I2C is a receiver
- 1 I2C is a transmitter

RXACKn This bit shows the state of ACK signal
- 0 No ACK is received
- 1 ACK is received at ninth SCLn period

NOTES:

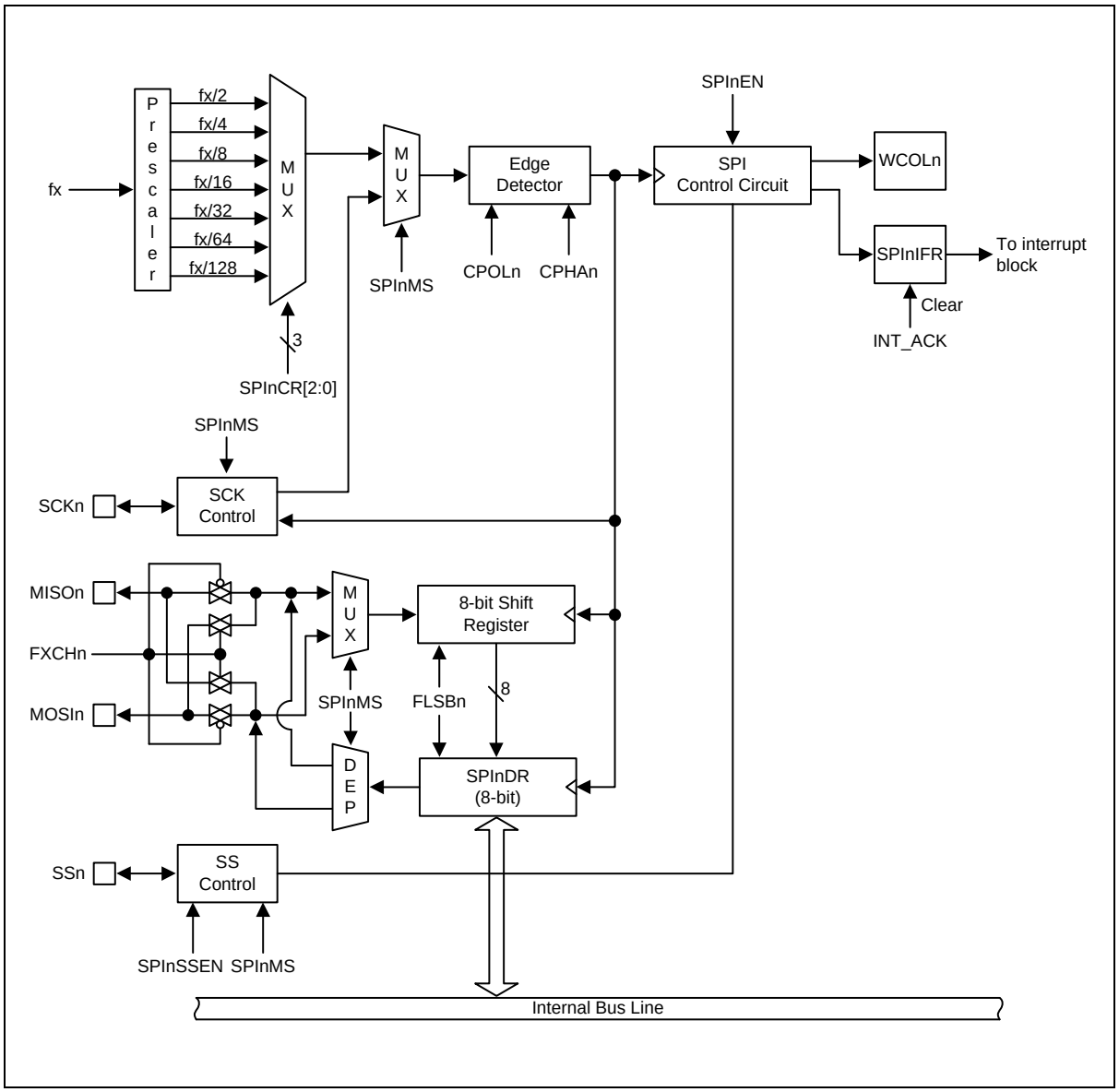
1. These bits can be the source of interrupt.
2. When an I2C interrupt occurs except for STOP mode, the SCLn line is hold LOW. To release SCLn, write arbitrary value to I2CnSR. When I2CnSR is written, The TENDn, STOPDn, SSELn, MLOSTn, and RXACKn bits are cleared.

15 SPI 0

There is serial peripheral interface (SPI) one channel in A96L116. The SPI allows synchronous serial data transfer between the external serial devices. It can do Full-duplex communication by 4-wire (MOSIn, MISOn, SCKn, SSn), support master/slave mode, can select serial clock (SCKn) polarity, phase and whether LSB first data transfer or MSB first data transfer.

15.1 Block Diagram

Figure 57. SPIn Block Diagram (where n = 0)



15.2 Data Transmit and Receive Operation

Users can use SPI for serial data communication by following the procedure below:

1. Select SPI operation mode (master/slave, polarity, phase) by control register SPInCR.
2. When the SPI is configured as a Master, it selects a Slave by SSn signal (active low). When the SPI is configured as a Slave, it is selected by SSn signal incoming from master device.
3. When the user writes a byte to the data register SPInDR, SPI will start an operation.
4. In this time, if the SPI is configured as a Master, serial clock will come out of SCKn pin. And Master shifts the eight bits into the Slave (transmit), Slave shifts the eight bits into the Master at the same time (receive). If the SPI is configured as a Slave, serial clock will come into SCKn pin. And Slave shifts the eight bits into the Master (transmit), Master shifts the eight bits into the Slave at the same time (receive).
5. When transmit/receive is done, SPInIFR bit will be set. If the SPI interrupt is enabled, an interrupt is requested. And SPInIFR bit is cleared by hardware when executing the corresponding interrupt. If SPI interrupt is disable, SPInIFR bit is cleared when user read the status register SPInSR, and then access (read/write) the data register SPInDR.

15.3 SSn Pin Function

This section describes the SSn pin settings when using SPI as a Master or Slave.

1. When the SPI is configured as a Slave, the SSn pin can be configured as SSn function by PnFSR register. If LOW signal comes into SSn pin, the SPI logic is active. And if ‘HIGH’ signal comes into SSn pin, the SPI logic is stop. In this time, SPI logic will be reset, and invalidated any received data.
2. When the SPI is configured as a Master, the user can select a mode of the SSn pin by port direction register. If the SSn pin is configured as an output or input, the user can use a general I/O pin. If the SSn pin is configured as an alternative function, ‘HIGH’ signal must come into SSn pin to guarantee Master operation. If ‘LOW’ signal comes into SSn pin, the SPI logic interprets this as another master selecting the SPI as a slave and starting to send data to it. To avoid bus contention, MSB bit of SPInCR will be cleared and the SPI becomes a Slave and then, SPInIFR bit of SPnISR will be set, and if the SPI interrupt is enabled, an interrupt is requested.

NOTES:

1. Before SPInCR setting, the mode of SSn pin must be defined by PnIO and PnFSR registers.
2. If you don’t need to use SSn pin, clear the SPInSSEN bit of SPInSR. So, you can use it as a normal I/O. In this case, SSn signal is driven by ‘HIGH’ or ‘LOW’ internally. In other words, master is ‘HIGH’, salve is ‘LOW’.
3. When SSn pin is configured as SSn function, if ‘HIGH’ signal come into SSn pin, SS_HIGH flag bit will be set. And users can clear it by writing ‘0’.

15.4 SPI 0 Timing Diagram

Figure 58. SPIn Transmit and Receive Timing Diagram at CPHA = 0 (where n = 0)

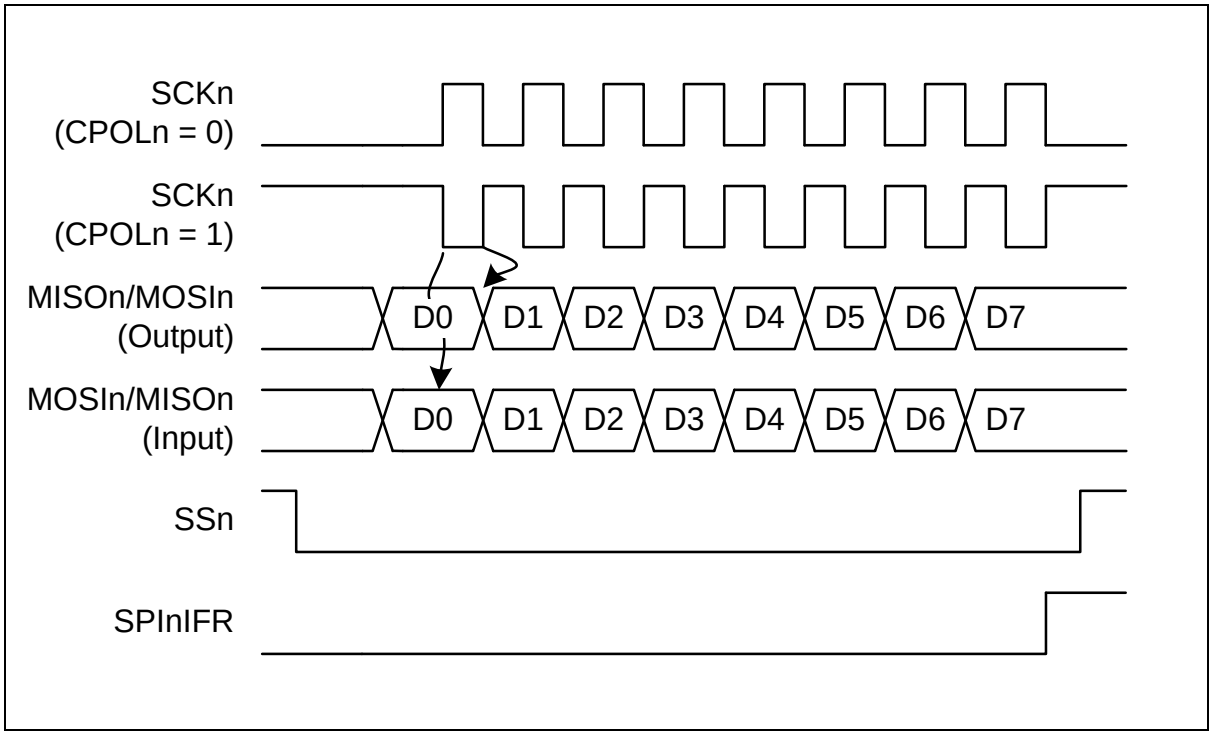
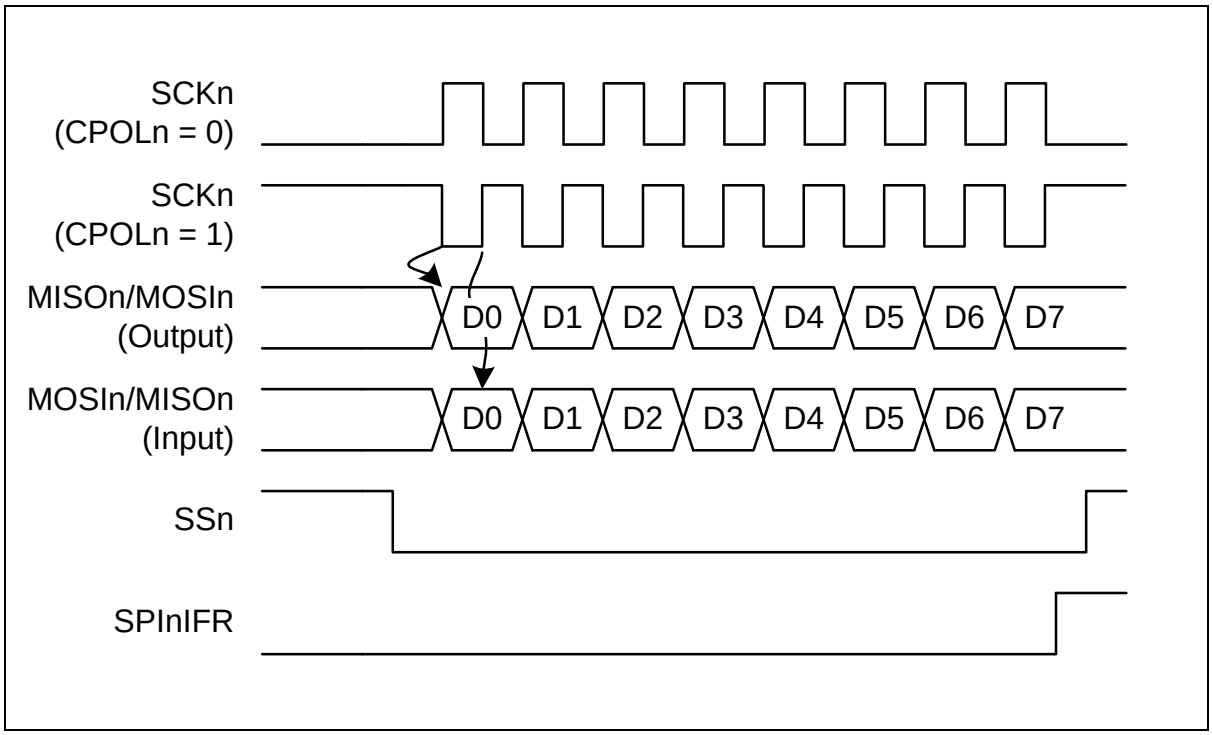


Figure 59. SPIn Transmit and Receive Timing Diagram at CPHA = 1 (where n = 0)



15.5 Register Map

Table 23. SPIn Register Map (where n = 0)

Name	Address	Direction	Default	Description
SPInDR	9DH	R/W	00H	SPI 0 Data Register
SPInSR	9EH	R/W	00H	SPI 0 Status Register
SPInCR	9CH	R/W	00H	SPI 0 Control Register

15.6 Register Description

SPInDR (SPIn Data Register): 9DH, where n = 0

7	6	5	4	3	2	1	0
SPInDR7	SPInDR6	SPInDR5	SPInDR4	SPInDR3	SPInDR2	SPInDR1	SPInDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

SPInDR[7:0] SPIn Data
When it is written a byte to this data register, the SPIn will start an operation.

SPInSR (SPIn Status Register): 9EH, where n = 0

7	6	5	4	3	2	1	0
SPInIFR	WCOLn	SS_HIGHn	–	FXCHn	SPInSSEN	–	–
R/W	R	R/W	–	R/W	R/W	–	–

Initial value: 00H

SPInIFR When SPIn Interrupt occurs, this bit becomes ‘1’. If SPIn interrupt is enabled, this bit is auto cleared by INT_ACK signal. And if SPIn Interrupt is disable, this bit is cleared when the status register SPInSR is read, and then access (read/write) the data register SPInDR. Writing “1” has no effect
- 0 SPIn interrupt no generation
- 1 SPIn interrupt generation

WCOLn This bit is set if any data are written to the data register SPInDR during transfer. This bit is cleared when the status register SPInSR is read, and then access (read/write) the data register SPInDR
- 0 No collision
- 1 collision

SS_HIGHn When the SSn pin is configured an alternative function, if “HIGH” signal comes into the pin, this flag bit will be set.
- 0 Cleared when ‘0’ is written
- 1 No effect when ‘1’ is written

FXCHn SPI 0 port function exchange control bit.
- 0 No effect
- 1 Exchange MOSIn and MISOn function

SPInSSEN This bit controls the SSn pin operation.
- 0 Disable
- 1 Enable (The corresponding pin should be configured an alternative function)

SPInCR (SPIn Control Register): 9CH, where n = 0

7	6	5	4	3	2	1	0
SPInEN	FLSBn	SPInMS	CPOLn	CPHAn	SPInDSCR	SPInSCR1	SPInSCR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

SPInEN This bit controls the SPIn operation

- 0 Disable SPIn operation
- 1 Enable SPIn operation

FLSBn This bit selects the data transmission sequence

- 0 MSB-first
- 1 LSB-first

SPInMS This bit selects whether Master or Slave mode

- 0 Slave mode
- 1 Master mode

CPOLn These two bits control the serial clock (SCKn) mode.

CPHAn Clock polarity (CPOLn) bit determines SCKn's value at idle mode.

Clock phase (CPHAn) bit determines if data are sampled on the leading or trailing edge of SCKn.

CPOLn	CPHAn	Leading edge	Trailing edge
0	0	Sample (Rising)	Setup (Falling)
0	1	Setup (Rising)	Sample (Falling)
1	0	Sample (Falling)	Setup (Rising)
1	1	Setup (Falling)	Sample (Rising)

SPInDSCR SPInSCR[1:0] These three bits select the SCKn rate of the device configured as a master. When DSCRn bit is written one, SCKn will be doubled in master mode.

SPInDSCR	SPInSCR1	SPInSCR0	SCKn frequency
0	0	0	$f_X/4$
0	0	1	$f_X/16$
0	1	0	$f_X/64$
0	1	1	$f_X/128$
1	0	0	$f_X/2$
1	0	1	$f_X/8$
1	1	0	$f_X/32$
1	1	1	$f_X/64$

16 Low Power UART 0

The A96L116 has a built-in 1-channel of low power UART module (Universal Asynchronous Receiver/Transmitter).

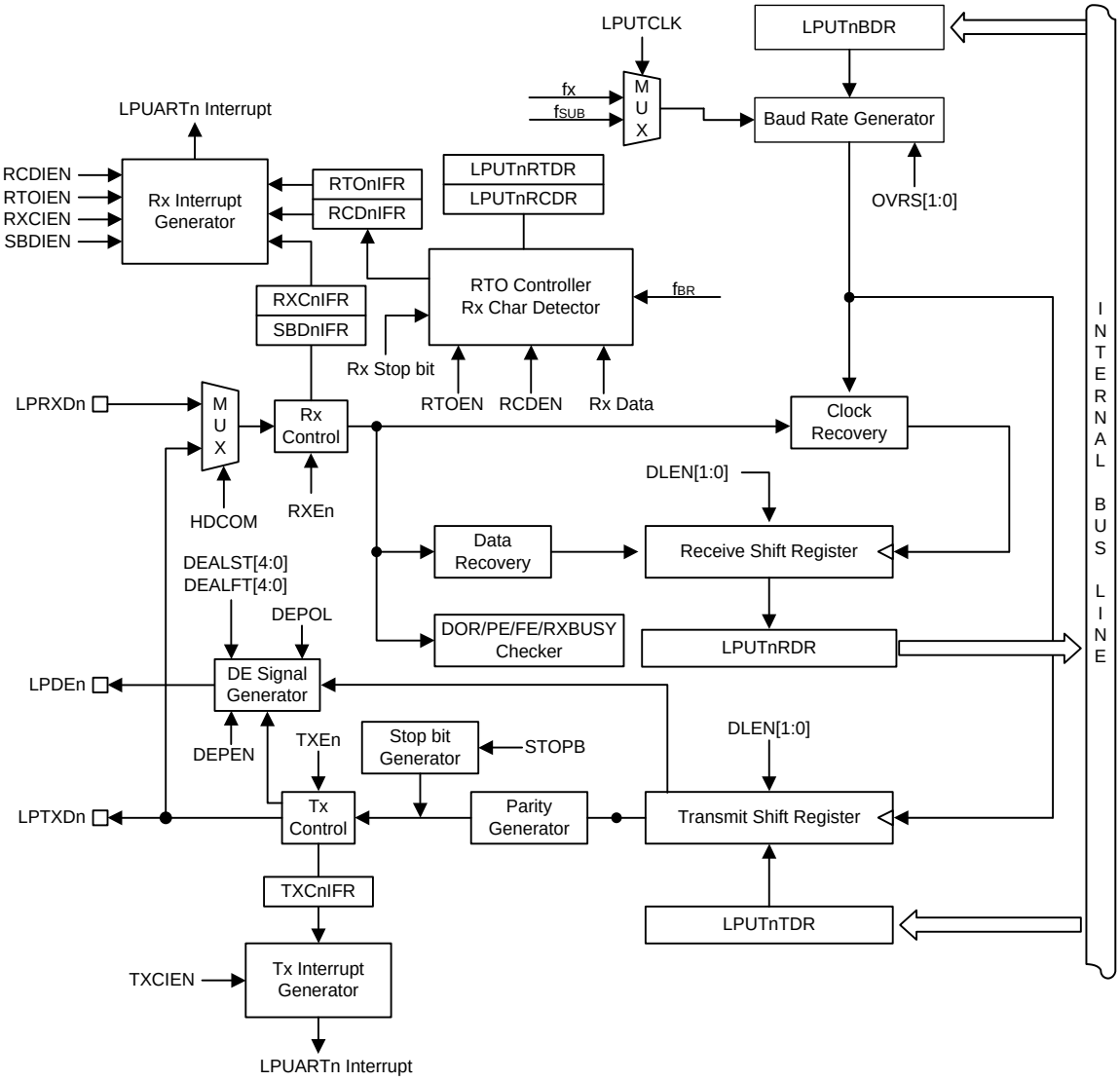
This LPUART supports asynchronous serial communication up to 9,600 bps in STOP mode when using a 32.768 kHz sub-oscillator. It also supports 1-wire half-duplex communications.

The LPUART of the A96L116 features the followings:

- Full-duplex and half-duplex operations
- Baud rate generator
- Supports serial frames with 5, 6, 7, or 8 data bits and 1 or 2 stop bits.
- Odd or even parity generation, and parity check supported by hardware.
- Supports receive character detection and receive time out function.
- Baud rate compensation function
- Supports up to 9,600 bps with 32.768 kHz sub-oscillator
- Data-OverRun Detection
- Framing Error Detection
- Double speed asynchronous communication mode.

16.1 Block Diagram

Figure 60. LPUARTn Block Diagram (where n = 0)



NOTES:

1. If the f_{SUB} (32.768 kHz) is used for clock of LPUARTn, the max. baud rate may be up to 9,600 bps.
2. The data to be transmitted should be written to the LPUTnTDR register after checking that the TXCnIFR bit is set to '1'.

16.2 Functional Description

The LPUART block comprises a clock generator, a transmitter, and a receiver. The clock generation logic consists of a baud rate generator. The transmitter consists of a write buffer, a serial shift register, a parity generator, and a control logic.

The receiver is the most complex part of the low power UART module due to its clock and data recovery units. The recovery unit is used for asynchronous data reception. In addition, the receiver has a parity checker, a shift register, and a control logic. The receiver supports the same frame formats as the transmitter and can detect frame errors, data overrun and parity errors.

16.2.1 LPUART Clock Generation

The clock generation logic generates clocks for the transmitter and the receiver. The LPUART baud rate generator supports three modes of clock operation, which are 16 oversampling mode, 8 oversampling mode, and only 1 sampling mode. Only 1 sampling mode can be used with SX-tal (32.768 kHz).

Following table shows equations for calculating the baud rate (in bps)

Table 24. Equations for Calculating Baud Rate Register Setting

Operating Mode	Equation for Calculating Baud Rate
16 oversampling mode (OVRS[1:0]=00)	Baud Rate = f_{LPUART} / (16 × (LPUTnBDR + 1))
8 oversampling mode (OVRS[1:0]=01)	Baud Rate = f_{LPUART} / (8 × (LPUTnBDR + 1))
Only 1 sampling mode (OVRS[1:0]=10)	Baud Rate = f_{LPUART} / (LPUTnBDR + 1)

16.2.2 LPUART Baud Rate Compensation

The baud rate compensation is used to optimize the precision in each bit. There is a sign (BCPS bit of LPUTnBCP register) bit to define the positive or negative compensation in each bit. If the sign bit is ‘0’, one clock of f_{LPUART} will be appended to the compensated bit. If the sign bit is ‘1’, one clock of f_{LPUART} will be taken out from the compensated bit.

There are nine bits to define whether relative compensation is required for each bit. The bits are BCP[7:0] for data and BCP8 for parity.

Example

1. f_{LPUART} = 32.768 kHz, No oversampling, Baud rate = 9,600 bps

   32.768 kHz / (1 × 9,600) = 3.413, LPUTnBDR = 3 − 1 = 2, and “Baud rate clock”/bit = 3 × 1

   So, “Clock error”/bit: 3.413 x 1 − 3 x 1 = 0.413 clock → “1 clock compensation”/bit if a BCPx bit is ‘1’.

   The result is that the sign bit, BCPS, is ‘0’ for positive compensation and the baud rate compensation bits, BCP[8:0], are “010100101”. (CEPB: “clock error”/bit).

Table 25. Baud Rate Compensation Example 1

Rx/Tx bit	BCPx bit	Clock Error	Compensation bit	Final clock error
Start bit	–	−0.413 (CEPB)	x	−0.413
D0	bit 0	−0.827 (CEPB+ before compensation)	1	0.173
D1	bit 1	−0.240 (CEPB+ before compensation)	0	−0.240
D2	bit 2	−0.653 (CEPB+ before compensation)	1	0.347
D3	bit 3	−0.067 (CEPB+ before compensation)	0	−0.067
D4	bit 4	−0.480 (CEPB+ before compensation)	0	−0.480
D5	bit 5	−0.893 (CEPB+ before compensation)	1	0.107
D6	bit 6	−0.307 (CEPB+ before compensation)	0	−0.307
D7	bit 7	−0.720 (CEPB+ before compensation)	1	0.280
Parity bit	bit 8	−0.133 (CEPB+ before compensation)	0	−0.133

2. f_{LPUART} = 32.768 kHz, No oversampling, Baud rate = 2,400 bps

   32.768 kHz/(1 x 2,400) = 13.653, LPUTnBDR = 14 − 1 = 13, and “Baud rate clock”/bit = 14 × 1

   So, “Clock error”/bit: 13.653 × 1 − 14 × 1 = −0.347 clock → “1 clock compensation”/bit if a BCPx bit is ‘1’.

   The result is that the sign bit, BCPS, is ‘1’ for negative compensation and the baud rate compensation bits, BCP[8:0], are “001001001”. (CEPB: “clock error”/bit).

Table 26. Baud Rate Compensation Example 2

Rx/Tx bit	BCPx bit	Clock Error	Compensation bit	Final clock error
Start bit	–	+0.347 (CEPB)	x	0.347
D0	bit 0	0.693 (CEPB+ before compensation)	1	−0.307
D1	bit 1	0.040 (CEPB+ before compensation)	0	0.040
D2	bit 2	0.387 (CEPB+ before compensation)	0	0.387
D3	bit 3	0.733 (CEPB+ before compensation)	1	−0.267
D4	bit 4	0.080 (CEPB+ before compensation)	0	0.080
D5	bit 5	0.427 (CEPB+ before compensation)	0	0.427
D6	bit 6	0.773 (CEPB+ before compensation)	1	−0.227
D7	bit 7	0.120 (CEPB+ before compensation)	0	0.120
Parity bit	bit 8	0.467 (CEPB+ before compensation)	0	0.467

16.2.3 LPUART Interface Data Format

A serial frame is defined to be composed of one character of data bits with synchronization bits (start and stop bits) and an optional parity bit for error detection.

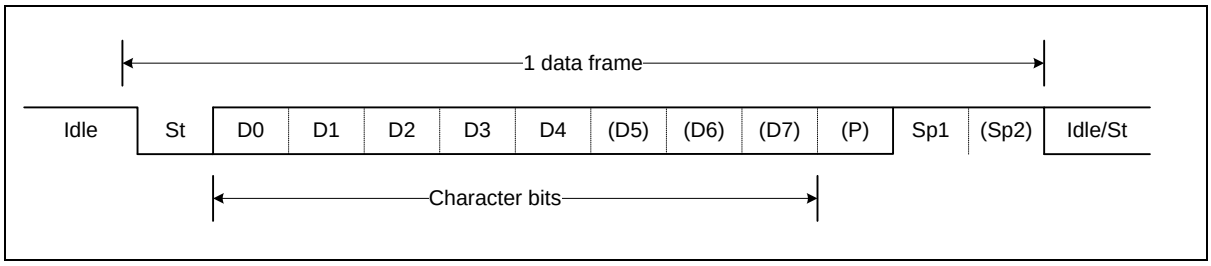
The LPUART supports all 24 combinations of the following as valid frame formats.

- 1 start bit
- 5, 6, 7, or 8 data bits
- No, even, or odd parity bit.
- 1 or 2 stop bits.

A frame starts with the start bit followed by the least significant data bit (LSB). Then the next data bits, up to eight, follow, ending with the most significant bit (MSB). If parity function is enabled, the parity bit is inserted between the last data bit and the stop bit.

A high-to-low transition on data pin is considered as start bit. When a complete frame is transmitted, it can be directly followed by a new frame, or the communication line can be set to an idle state. The idle state means high state of data pin. The following figure shows the possible combinations of the frame formats. Bits inside round brackets are optional.

Figure 61. Frame Format



One data frame consists of the following bits:

- Idle: No signal on communication line (LPTXD/LPRXD)
- St: Start bit (Low)
- Dm: Data bits (0 ~ 7)
- P: Parity bit (even parity, odd parity, no parity)
- Sp: Stop bit (1 bit or 2 bits)

The frame format is set by configuring DLEN[1:0], PSEL, PEN, and STOPB bits in the LPUTnCR1 register. The transmitter and the receiver use the same values.

16.2.4 LPUART Interface Parity Bit

The parity bit is calculated by doing an exclusive-OR of all data bits. If odd parity is used, the result of the exclusive-OR is inverted. The parity bit is located between the last data bit and first stop bit of a serial frame.

- Peven = Dm-1 ^ … ^ D3 ^ D2 ^ D1 ^ D0 ^ 0
- Podd = Dm-1 ^ … ^ D3 ^ D2 ^ D1 ^ D0 ^ 1
- Peven: Parity bit using even parity
- Podd: Parity bit using odd parity
- Dm: Data bit n of the character

16.2.5 LPUART Transmitter

The LPUART transmitter is enabled by setting the TXE bit in LPUTnCR1 register. When the transmitter is enabled, the LPTXD pin should be set to LPTXD function for the serial output pin by the GPIO registers.

Baud-rate, operation mode and frame format must be set up before doing any transmission.

16.2.6 LPUART Sending Tx Data

A data transmission is initiated by loading data to the transmit data register (LPUTnTDR register). The data to be written in the transmit data register is moved to the shift register when the shift register is ready to send a new frame.

The shift register is loaded with the new data if it is in idle state or immediately after the last stop bit of the previous frame is transmitted. When the shift register is loaded to the new data, it will transfer one complete frame according to the settings of control registers. (n = 0).

16.2.7 LPUART Parity Generator

The parity generator calculates parity bit for the serial frame data to be sent. When the parity bit is enabled (PEN = 1), the transmitter control logic inserts the parity bit between the last data bit and the first stop bit of the frame to be sent.

16.2.8 LPUART Receiver

The LPUART receiver is enabled by setting the RXE bit in the LPUTnCR1 register. When the receiver is enabled, the LPRXD pin should be set to LPRXD function for the serial input pin by the GPIO registers.

Baud-rate, operation mode, and frame format must be set before the serial reception.

16.2.9 LPUART Receiving RX Data

The receiver starts data reception when it detects a valid start bit (LOW) on LPRXD pin. Each bit after start bit is sampled at predefined baud-rate, and shifted into the receive shift register until the first stop bit of a frame is received.

Even if there is the second stop bit in the frame, the second stop bit is ignored by the receiver. That is, receiving the first stop bit means that a complete serial frame is presented in the receiver shift register and contents of the shift register are to be moved into the receive data register.

16.2.10 LPUART Parity Checker

If the parity bit is enabled (PEN = 1), the Parity Checker calculates the parity of the data bits in incoming frame and compares the result with the parity bit from the received serial frame.

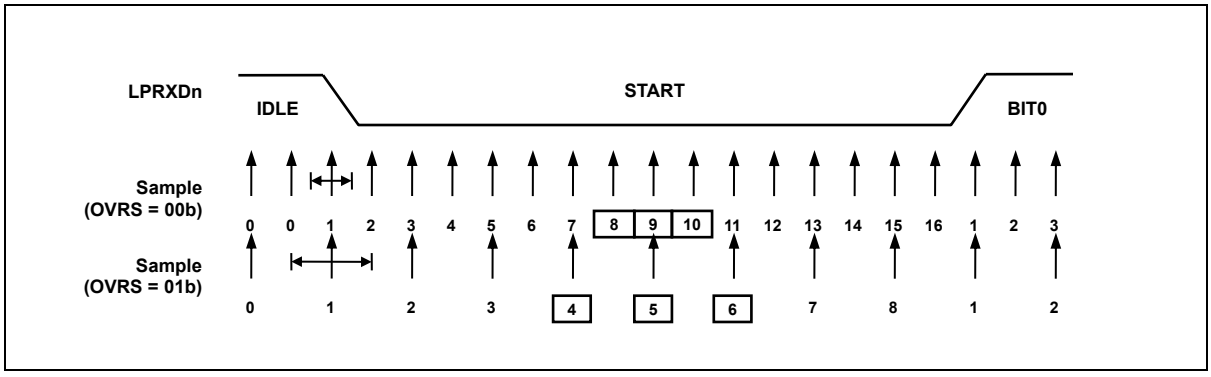
16.2.11 LPUART Data Reception

To receive data frame, the receiver includes a clock and data recovery unit. The clock recovery logic is used for synchronizing the internally generated baud-rate clock to the incoming asynchronous serial frame on the LPRXD pin.

The data recovery logic samples and filters the incoming bits with a low pass filter and removes the noise of the receive pin.

In this Figure, illustrates the sampling process of a start bit of an incoming frame. The sampling rate is 16 times the baud rate in 16-oversampling mode and 8 times the baud rate for 8-oversampling mode. The horizontal arrows show the synchronization variation due to the asynchronous sampling process. Note that larger time variation is seen when using 8-oversampling mode.

Figure 62. Start Bit Sampling



When the receiver is enabled (RXEn=1), the clock recovery logic tries to find a high-to-low transition on the LPRXD line, the start bit condition. After detecting high to low transition on the line, the clock recovery logic uses samples 8, 9 and 10 for 16-oversampling mode to detect whether valid start bit is received.

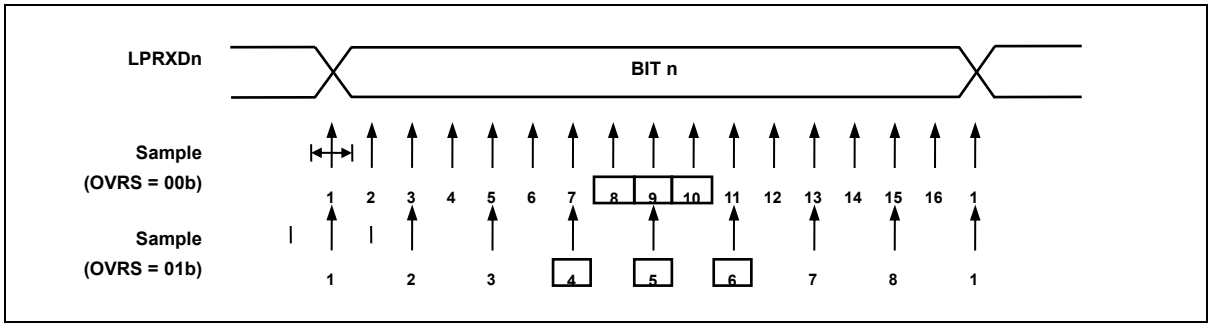
If more than two samples have logical low level, it is considered that a valid start bit is detected, and the internally generated clock is synchronized to the incoming data frame. Then the data recovery can begin. The synchronization process is repeated for each start bit.

As described above, when the receiver clock is synchronized to the start bit, the data recovery can begin. Data recovery process is almost the same as clock recovery process. The data recovery logic samples each incoming bit 16 times for 16-oversampling mode and 8 times for 8-oversampling mode, and uses sample 8, 9 and 10 to decide data value. If more than two samples have low levels, the received bit is considered as a logic '0' and if more than two samples have high levels, the received bit is considered as a logic '1'.

The data recovery process is then repeated until a complete frame is received, including the first stop bit. The decided bit value is stored in the receive shift register in order.

Note that the receiver only uses the first stop bit of a frame. Internally, after receiving the first stop bit, the receiver is in idle state and waits to find the start bit.

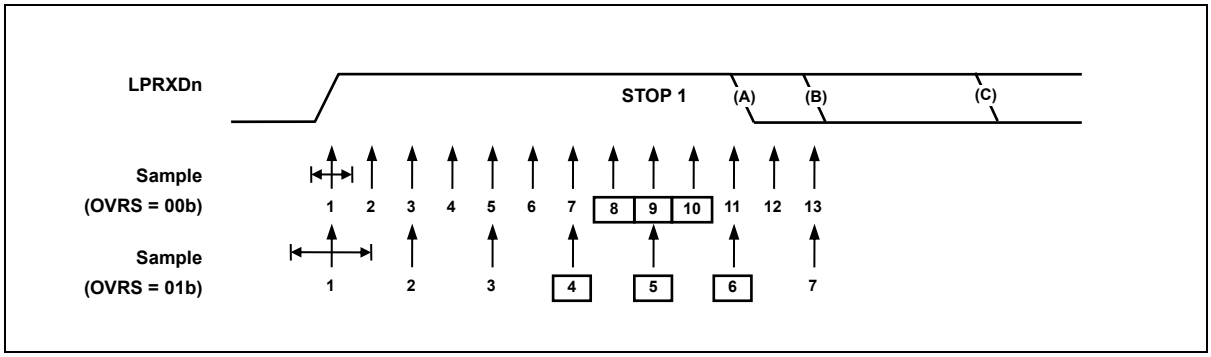
Figure 63. Sampling of Data and Parity Bit



The process for detecting stop bit is the same as clock and data recovery process. That is, if two or more samples of three center values have high level, correct stop bit is detected, or else a frame error (FE) flag is set.

After deciding whether the first stop bit is valid or not, the receiver goes to idle state and monitors the LPRXDn line to check whether a valid high to low transition is detected (start bit detection).

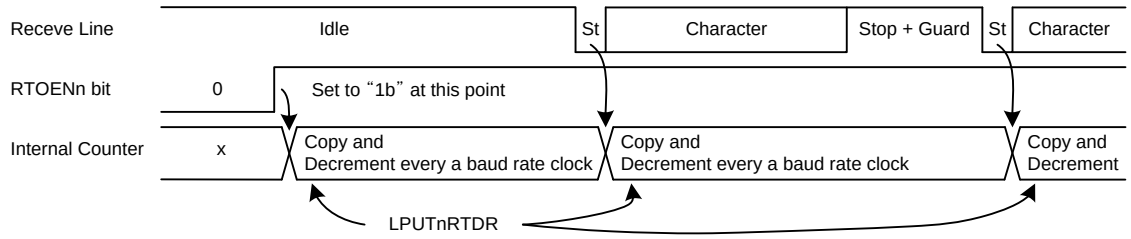
Figure 64. Stop Bit Sampling and Next Start Bit Sampling



16.2.12 LPUART Receive Time Out Function

The receive time out function is used for checking a frame finish. This function is to count time with baud rate unit between the last start bit and a new start bit, and between setting the RTOEN bit of the LPUTnCR1 register and a new start bit. The LPUTnRTDRH/L register should have duration time value before using the receive time out function.

Figure 65. Receive Time Out Function



NOTES:

1. The value of LPUTnRTDRH/L register is transferred to an internal counter when the RTOEN bit is set to "1b" or when a start bit is received.
2. If the RTOEN bit is set to "1b", the internal counter counts down until underflow.
3. The RTOnIFR bit of LPUTnISRL register is set to "1b" on the internal counter underflow and the receive time out function is finished. So, to re-use the receive time out function, the RTOEN bit must be cleared to "0b" and then set to "1b" again.

16.2.13 1-wire Half-duplex Communication

1-wire half-duplex mode is selected by configuring HDCOM bit in the LPUTnCR1 register. The TXD and the RXD lines are internally connected, the RXD pin is not in use, and the TXD pin is always an input when not transmitting. So, the TXD pin must be configured to open-drain with an external pull-up resistor.

16.3 Register Map

Table 27. LPUARTn Register Map (where n = 0)

Name	Address	Direction	Default	Description
LPUTnCR0	E2H	R/W	00H	LPUARTn Control Register 0
LPUTnCR1	E3H	R/W	00H	LPUARTn Control Register 1
LPUTnCR2	E4H	R/W	00H	LPUARTn Control Register 2
LPUTnCR3	E5H	R/W	00H	LPUARTn Control Register 3
LPUTnCR4	E6H	R/W	00H	LPUARTn Control Register 4
LPUTnIER	E7H	R/W	00H	LPUARTn Interrupt Enable Register
LPUTnBCPH	D3H	R/W	00H	LPUARTn Baud Rate Compensation High Register
LPUTnBCPL	D2H	R/W	00H	LPUARTn Baud Rate Compensation Low Register
LPUTnRTDRH	D5H	R/W	FFH	LPUARTn Receive Time Out Data High Register
LPUTnRTDRL	D4H	R/W	FFH	LPUARTn Receive Time Out Data Low Register
LPUTnRCDR	D6H	R/W	00H	LPUARTn Receive Character Detection Data Register
LPUTnDLY	D7H	R/W	00H	LPUARTn Tx Delay Time Data Register
LPUTnISRH	DBH	R/W	00H	LPUARTn Interrupt flag and Status High Register
LPUTnISRL	DAH	R/W	02H	LPUARTn Interrupt flag and Status Low Register
LPUTnRDR	DCH	R	00H	LPUARTn Receive Data Register
LPUTnTDR	DDH	R/W	00H	LPUARTn Transmit Data Register
LPUTnBDR	DEH	R/W	FFH	LPUARTn Baud Rate Data Register

16.4 Register Description

LPUTnCR0 (LPUARTn Control Register 1): E2H, Where n = 0

7	6	5	4	3	2	1	0
PEN	STKPEN	PSEL	LPUTCLK	–	DLEN1	DLEN0	STOPB
R/W	R/W	R/W	R/W	–	R/W	R/W	R/W

Initial value: 00H

PEN Parity Enable bit
- 0 Disable parity bit generation and detection
- 1 Enable parity bit generation and detection

STKPEN Stick Parity Enable bit
- 0 Disable stick parity
- 1 Enable stick parity

NOTE:
1. On PEN=1 and STKPEN=1, The parity bit is '0' if PSEL=0 and 1 if PSEL=1

PSEL Parity Selection bit
- 0 Odd parity (Odd number of logic '1')
- 1 Even parity (Even number of logic '1')

LPUTCLK LPUARTn Clock Selection bit
- 0 Select fx for LPUARTn clock
- 1 Select f_{SUB} for LPUARTn clock

DLEN Data Length Selection bits
- 0 0 5-bit (Start, D0 to D4, Parity or not, Stop1, Stop2 or not)
- 0 1 6-bit (Start, D0 to D5, Parity or not, Stop1, Stop2 or not)
- 1 0 7-bit (Start, D0 to D6, Parity or not, Stop1, Stop2 or not)
- 1 1 8-bit (Start, D0 to D7, Parity or not, Stop1, Stop2 or not)

STOPB Stop bit
- 0 1-stop bit
- 1 2-stop bit

LPUTnCR1 (LPUARTn Control Register 1): E3H, where n = 0

7	6	5	4	3	2	1	0
–	OVRS1	OVRS0	HDCOM	TXEn	RXEn	WAKEN	LPUEN
–	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

OVRS[1:0] Oversampling Selection bits

OVRS1	OVRS0	Description
0	0	16-Oversampling
0	1	8-Oversampling
1	0	No oversampling (single sampling)
1	1	Not available (No oversampling)

HDCOM 1-wire Half-Duplex Communication

0	Normal operation
1	1-wire half-duplex communication (The TXD and RXD lines are internally connected, the RXD pin is not used, and the TXD pin is always an input when not transmitting. So, the TXD pin must be configured to open-drain with an external pull-up resistor)

TXEn Enable the Transmitter unit

0	Transmitter is disabled
1	Transmitter is enabled

RXEn Enable the Receiver unit

0	Receiver is disabled
1	Receiver is enabled

WAKEN Wake-up Function bit in Stop Mode. The LPUART clock to wake-up from stop mode must be selected as f_{SUB} by the LPUTCLK bit. This bit should be set just before entering stop mode and cleared on exit.

0	Disable wake-up function in stop mode
1	Enable wake-up function in stop mode

NOTE:

1. If f_{SUB} is clock of LPUARTn, the f_{SUB} shouldn't be off in stop mode

LPUEN Low Power UARTn Enable bit. This bit can be cleared to "0b" during the corresponding TXEn and RXEn bits are all "0b"

0	Disable LPUARTn block
1	Enable LPUARTn block

NOTE:

1. If this bit is cleared, the LPUARTn current operations are discarded, the configuration is kept, and all the status flags are set to reset values.

LPUTnCR2 (LPUARTn Control Register 2): E4H (n = 0)

7	6	5	4	3	2	1	0
–	–	–	DEALST4	DEALST3	DEALST2	DEALST1	DEALST0
–	–	–	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

DEALST[4:0]	DE Pin Active Level Start Time bits. The range is 0x00 to 0x1f. These bits define the time in LPUARTn clock from the active level of DE signal to the beginning of the start bit.

LPUTnCR3 (LPUARTn Control Register 3): E5H (n = 0)

7	6	5	4	3	2	1	0
–	–	–	DEALFT4	DEALFT3	DEALFT2	DEALFT1	DEALFT0
–	–	–	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

DEALFT[4:0]	DE Pin Active Level Finish Time bits. The range is 0x00 to 0x1f. These bits define the time in LPUARTn clock from the active level of DE signal to the beginning of the start bit.

LPUTnCR4 (LPUARTn Control Register 4): E6H (n = 0)

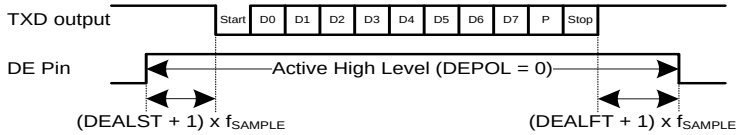
7	6	5	4	3	2	1	0
–	–	–	DEPOL	DEPEN	–	RCDEN	RTOEN
–	–	–	R/W	R/W	–	R/W	R/W

Initial value: 00H

DEPOL DE Pin Polarity

0 Active high level. The DE pin is a high level during transmit a frame, else low level.

1 Active low level. The DE pin is a low during transmit a frame, else high level.



Where $f_{SAMPLE}=f_{LPUARTn}/(LPUTnBDR[7:0] + 1)$

NOTE:

A TXCnIFR bit will be set to “1b” at stop bit and the transmit of next character may start after the end of active level.

DEPEN DE Pin Function Enable bit

0 Disable DE pin function

1 Enable DE pin function

RCDEN Receive Character Detection Function Enable bit. This function is to compare the value of LPUTnRCDR register with the value just received

0 Disable receive detection function

1 Enable receive detection function

RTOEN Receive Time Out Function Enable bit. This function is to count time with baud rate units from the leading edge of a start bit to a new start bit. The receive time out controller counts down from the value of LPUTnRTDR register every start bit and set this bit. The RTOnIFR bit is set to “1b” at the counter underflow. The counter clock is a baud-rate bit unit.

0 Disable receive time out function

1 Enable receive time out function

LPUTnIER (LPUARTn Interrupt Enable Register): E7H (n = 0)

7	6	5	4	3	2	1	0
–	–	–	RCDIEN	RTOIEN	SBDIEN	TXCIEN	RXCIEN
–	–	–	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

RCDIEN Receive Character Detection Interrupt Enable bit. On stop mode, the receive character detection can wake-up system.
- 0 Disable receive character detection interrupt.
- 1 Enable receive character detection interrupt.

RTOIEN Receive Time Out Interrupt Enable bit.
- 0 Disable receive time out interrupt.
- 1 Enable receive time out interrupt.

SBDIEN Start Bit Detection Interrupt Enable bit in Stop Mode. On stop mode, the detection of start bit can wake-up system.
- 0 Disable start bit detection interrupt.
- 1 Enable start bit detection interrupt.

TXCIEN Transmit Complete Interrupt Enable bit.
- 0 Disable transmit complete interrupt.
- 1 Enable transmit complete interrupt.

RXCIEN Receive Data Register Not Empty Interrupt Enable bit. On deep sleep mode, it can wake-up system if there is a received character.
- 0 Disable receive data not empty interrupt.
- 1 Enable receive data not empty interrupt.

LPUTnISRH (LPUARTn Interrupt Flag and Status High Register): DBH (n = 0)

7	6	5	4	3	2	1	0
–	–	–	–	DOR	FE	PE	RXBUSY
–	–	–	–	R/W	R/W	R/W	R

Initial value: 00H

DOR Data-Overrun bit. This bit is set when the receive shift register is transferred to the LPUTnRDR register while the RXCnIFR=1. The data in the shift register are ignored. This bit must be cleared by software to receive new data. This bit is cleared by writing ‘0’ to the bit.
- 0 No data overrun
- 1 Data overrun detected

FE Frame Error bit. This bit is set when the received data have not a valid stop bit (that is, the stop bit following the last data bit is detected as “0b”). The bit will be cleared by hardware if new data are received. This bit is cleared by writing ‘0’ to the bit.
- 0 No frame error
- 1 Frame error detected

PE Parity Error bit. This bit is set when the received data has a parity error on parity enable. The bit will be cleared by hardware if new data are received. This bit is cleared by writing ‘0’ to the bit.
- 0 No parity error
- 1 Parity error detected

RXBUSY RXD Line Busy bit. This bit is set at a start bit and reset at the end of the reception.
- 0 Receive line (RXD) is not busy
- 1 Reception on going

LPUTnISRL (LPUARTn Interrupt Flag and Status Low Register): DAH (n = 0)

7	6	5	4	3	2	1	0
–	–	–	RCDnIFR	RTOnIFR	SBDnIFR	TXCnIFR	RXCnIFR
–	–	–	R/W	R/W	R/W	R/W	R/W

Initial value: 02H

RCDnIFR — Receive Character detection Interrupt Flag. This bit is set to ‘1’ when the value of LPUTnRCDR register matches the value received in the non-error state of frame and parity. On match of them, the bit may be set even if data overrun occurs. The flag is cleared only by writing ‘0’ to the bit. So, the flag should be cleared by software.

0 No request occurred

1 Request occurred

RTOnIFR — Receive Time Out Interrupt Flag. This bit is set to ‘1’ at the counter underflow of the receive time out controller. The flag is cleared only by writing ‘0’ to the bit. So, the flag should be cleared by software.

0 No request occurred

1 Request occurred

SBDnIFR — Start Bit Detection Interrupt Flag. This bit is set to ‘1’ when a start bit is detected in stop mode. The flag is cleared only by writing ‘0’ to the bit. So, the flag should be cleared by software.

0 No request occurred

1 Request occurred

TXCnIFR — Transmit Complete Interrupt Flag. This flag is set to ‘1’ when the data in the transmit shift register has been shifted out. The flag is cleared only by writing ‘0’ to the bit. So, the flag should be cleared by software.

0 No request occurred

1 Request occurred

RXCnIFR — Receive Data Register Not Empty Interrupt Flag. This bit is set to ‘1’ when the data in the receive shift register has been transferred to the LPUTnRDR register. The bit is cleared by a read to the LPUTnRDR register. The flag is cleared only by writing ‘0’ to the bit. So, the flag should be cleared by software.

0 No request occurred

1 Request occurred

LPUTnRTDRH (LPUARTn Receive Time Out Data High Register): D5H (n = 0)

7	6	5	4	3	2	1	0
RTOD15	RTOD14	RTOD13	RTOD12	RTOD11	RTOD10	RTOD9	RTOD8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

RTOD[15:8] LPUARTn Receive Time Out Data High Byte

LPUTnRTDRL (LPUARTn Receive Time Out Data Low Register): D4H (n = 0)

7	6	5	4	3	2	1	0
RTOD7	RTOD6	RTOD5	RTOD4	RTOD3	RTOD2	RTOD1	RTOD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

RTOD[7:0] LPUARTn Receive Time Out Data Low Byte

This RTOD[15:0] bits are count time with baud rate unit between the last start bit and a new start bit. The time is (RTOD[15:0] + 1) × “baud rate clock period”, that is (RTOD[15:0] + 1) × {(LPUTnBDR + 1) / f_{LPUART}}.

LPUTnRCDR (LPUARTn Receive Character Detection Data Register): D6H (n = 0)

7	6	5	4	3	2	1	0
RCDD7	RCDD6	RCDD5	RCDD4	RCDD3	RCDD2	RCDD1	RCDD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

RCDD[7:0] LPUARTn Receive Character Detection Data

LPUTnDLY (LPUARTn Tx Delay Time Data Register): D7H (n = 0)

7	6	5	4	3	2	1	0
DLYD7	DLYD6	DLYD5	DLYD4	DLYD3	DLYD2	DLYD1	DLYD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

DLYD[7:0] LPUARTn Tx Delay Data. This register is used for transmit delay time between the last stop bit and the next start bit with baud rate unit.

The data in the LPUTnTDR register will be transferred to the transmit shift register after delay time.

Delay time: DLYD[7:0] × “baud rate clock period”, that is DLYD[7:0] × {(LPUTnBDR + 1) / f_{LPUART}}. No delay on DLYD[7:0] = 0.

LPUTnRDR (LPUARTn Receive Data Register): DCH (n = 0)

7	6	5	4	3	2	1	0
RDATA7	RDATA6	RDATA5	RDATA4	RDATA3	RDATA2	RDATA1	RDATA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

RDATA[7:0] LPUARTn Receive Data Byte

A receive shift register is moved to this register after stop bit.

LPUTnTDR (LPUARTn Transmit Data Register): DDH, (n = 0)

7	6	5	4	3	2	1	0
TDATA7	TDATA6	TDATA5	TDATA4	TDATA3	TDATA2	TDATA1	TDATA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

TDATA[7:0] LPUARTn Transmit Data Byte

This register is moved to the transmit shift register after a previous character is completely shifted out.

LPUTnBDR (LPUARTn Baud Rate Data Register): DEH (n = 0)

7	6	5	4	3	2	1	0
BDATA7	BDATA6	BDATA5	BDATA4	BDATA3	BDATA2	BDATA1	BDATA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: FFH

BDATA[7:0] LPUARTn Baud Rate Data Byte

16 oversampling:

- Baud Rate=$f_{LPUARTn}$ / {16 × (BDATA[7:0] + 1)}
- BDATA[7:0] range: 0x0 to 0xFE

8 oversampling:

- Baud Rate=$f_{LPUARTn}$ / {8 × (BDATA[7:0] + 1)}
- BDATA[7:0] range: 0x0 to 0xFE

No oversampling: This can be used with SX-tal (32.768 kHz).

- Baud Rate=$f_{LPUARTn}$ / {(BDATA[7:0] + 1)}
- BDATA[7:0] range: 0x2 to 0xFE
- If this register is 0x02 on the no oversampling, the BCPS bit of the LPUTnBCPH register shouldn’t be set to ‘1’ for minus compensation

LPUTnBCPH (LPUARTn Baud Rate Compensation High Register): D3H (n = 0)

7	6	5	4	3	2	1	0
BCPS	–	–	–	–	–	–	BCP8
R/W	–	–	–	–	–	–	R/W

Initial value: 00H

BCPS Baud Rate Compensation Sign bit
- 0 Plus 1 clock for compensation
- 1 Minus 1 clock for compensation

BCP8 Baud Rate Compensation 8th bit
- 0 No compensation
- 1 1 clock compensation every sampling with sign bit.

NOTE:
1. The BCP8 bit is for parity bit.

LPUTnBCPL (LPUARTn Baud Rate Compensation Low Register): D2H (n = 0)

7	6	5	4	3	2	1	0
BCP7	BCP6	BCP5	BCP4	BCP3	BCP2	BCP1	BCP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

BCPx Baud Rate Compensation bit. X: 0 to 7
- 0 No compensation
- 1 1 clock compensation every sampling with sign bit.

NOTE:
1. The BCP[7:0] bit is for Data[7:0].

17 Flash CRC and Checksum Generator

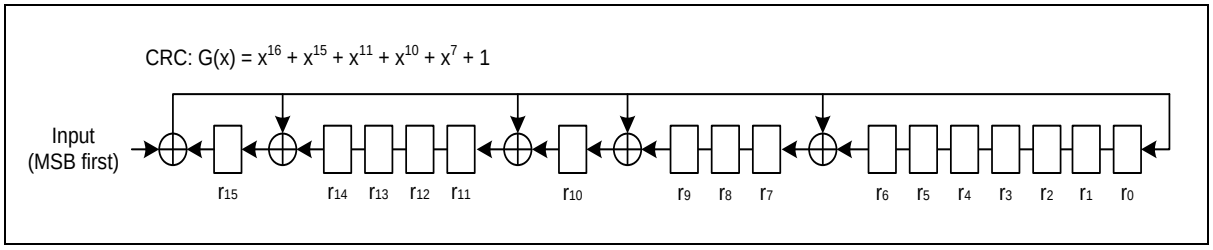
Flash CRC (Cyclic Redundancy Check) generator of A96L116 generates 16-bit CRC code bits from flash and a generator polynomial. The CRC code for each input data frame is appended to the frame.

Specifically, CRC-based technique is used to verify data transmission or storage integrity. In the scope of the functional safety standards, this technique offers a means of verifying the flash memory integrity. The Flash CRC generator helps compute a signature of software during runtime, to be compared with a reference signature.

The CRC generator has following features:

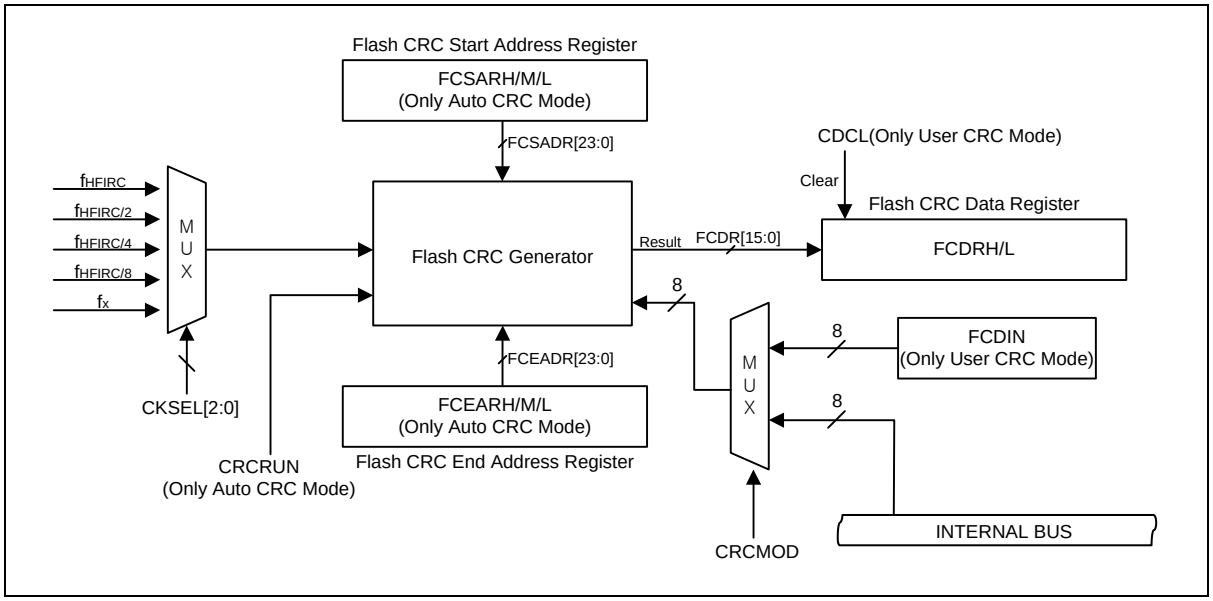
- Auto CRC and User CRC Mode
- CRC Clock: f_{HFIRC}, $f_{HFIRC}/2$, $f_{HFIRC}/4$, $f_{HFIRC}/8$ and fx (System clock)
- CRC-16 polynomial: 0x8C81 ($X^{16} + X^{15} + X^{11} + X^{10} + X^7 + 1$)

Figure 66. CRC-16 Polynomial Structure



17.1 Block Diagram

Figure 67. Flash CRC/Checksum Generator Block Diagram



17.2 Operation Procedure and Example Code of CRC and Checksum

The CRC operation procedure in Auto CRC/Checksum mode is described in the following list, and Figure 68 shows example program tip:

1. Global interrupt Disable (EA = 0)
2. Select Auto CRC/Checksum Mode and CRC.
3. Select CRC Clock.
4. Set CRC start address register (FCSARH/FCSARM/FCSARL).
5. Set CRC end address register (FCEARH/FCEARM/FCEARL).
6. CRC operation starts (CRCRUN = 1).
7. Read the CRC result.
8. Global interrupt Enable (EA = 1)

Figure 68. Program Tip for CRC Operation in Auto CRC/Checksum Mode

```
1             //**** Global interrupt Disable
2             EA = 0;
3
4             //**** Flash CRC Auto CRC/Checksum Mode and CRC
5             FCCR &= _0101_1111;
6
7             OSCCR &= _1111_1011;       // IRC Enable
8             FCCR &= _1111_0001;                // CRC clk = fIRC/1
9
10            //**** CRC start address set
11            FCSARH = 0x00;
12            FCSARM = 0x00;
13            FCSARL = 0x00;
14
15            //**** CRC end address set
16            FCEARH = 0x00;
17            FCEARM = 0x1F;
18            FCEARL = 0xFF;
19
20            //**** CRC start
21            FCCR |= _0000_0001;
22            _nop_();          //Dummy instruction, This instruction must be needed.
23            _nop_();          //Dummy instruction, This instruction must be needed.
24            _nop_();          //Dummy instruction, This instruction must be needed.
25
26            //**** Read CRC result
27            Temp0 = FCDRH;
28            Temp1 = FCDRL;
29
30            //**** Global interrupt Enable
31            EA = 1;
```

NOTES:

1. Three or more NOP instructions must immediately follow the CRC start operation in auto CRC/Checksum mode.
2. During a CRC operation (when CRCRUN bit is Running state) in auto CRC/Checksum mode, the CPU is hold and the global interrupt is on the disable-state regardless of the IE.7 (EA) bit. But should be set the global interrupt is disabled (EA = 0) before the CRC operation is started in use auto CRC/Checksum mode, recommend.

The CRC operation procedure in User CRC/Checksum mode is described in the following list, and Figure 69 shows example program tip:

1. Select User CRC/Checksum Mode and CRC.
2. Clear Flash CRC data register (FCDRH/FCDRL).
3. Read data from the flash memory.
4. Write the data to FCDIN Register.
5. Read the CRC result.

Figure 69. Program Tip for CRC Operation in User CRC/Checksum Mode

```
1         unsigned char code *rom_addr=0x0000;
2         unsigned int i=0;
3
4              FCCR |= _1000_0000;         // Flash CRC User CRC/Checksum Mode
5              FCCR &= _1101_1111;         // Flash CRC CRC Mode
6              FCCR |= _0100_0000;         // Flash CRC data register clear
7
8              for(i=0x0000; i <= 0x1FFF; i++)      // 0000H~1FFFH
9              {
10                      FCDIN = rom_addr[i];
11                      WDTCR |= _0010_0000;        // Clear WDT counter
12             }
13
14        //**** Read CRC result
15                Temp0 = FCDRH;
16                Temp1 = FCDRL;
```

The Checksum operation procedure in Auto CRC/Checksum mode is described in the following list, and Figure 70 shows example program tip:

1. Global interrupt Disable (EA = 0)
2. Select Auto CRC/Checksum Mode and Checksum.
3. Select CRC Clock.
4. Set CRC start address register (FCSARH/FCSARM/FCSARL).
5. Set CRC end address register (FCEARH/FCEARM/FCEARL).
6. CRC operation starts (CRCRUN = 1).
7. Read the Checksum result.
8. Global interrupt Enable (EA = 1)

Figure 70. Program Tip for Checksum Operation in Auto CRC/Checksum Mode

```
1         //**** Global interrupt Disable
2         EA = 0;
3
4         //**** Flash CRC Auto CRC/Checksum Mode and Checksum
5         FCCR &= _0111_1111;
6         FCCR |= _0010_0000;            // Checksum mode
7
8         OSCCR &= _1111_1011;           // IRC Enable
9         FCCR &= _1111_0001;            // CRC clk = fIRC/1
10
11        //**** Checksum start address set
12        FCSARH = 0x00;
13        FCSARM = 0x00;
14        FCSARL = 0x00;
15
16        //**** Checksum end address set
17        FCEARH = 0x00;
18        FCEARM = 0x1F;
19        FCEARL = 0xFF;
20
21        //**** Checksum start
22        FCCR |= _0000_0001;
23        _nop_();              //Dummy instruction, This instruction must be needed.
24        _nop_();              //Dummy instruction, This instruction must be needed.
25        _nop_();              //Dummy instruction, This instruction must be needed.
26
27        //**** Read Checksum result
28        Temp0 = FCDRH;
29        Temp1 = FCDRL;
30
31        //**** Global interrupt Enable
32        EA = 1;
```

NOTES:

1. Three or more NOP instructions must immediately follow the Checksum start operation in auto CRC/Checksum mode.
2. During a checksum operation (when CRCRUN bit is Running state) in auto CRC/Checksum mode, the CPU is hold and the global interrupt is on disable-state regardless of the IE.7 (EA) bit. But should be set the global interrupt is disabled (EA = 0) before the Checksum operation is started in use auto CRC/Checksum mode, recommend.

The Checksum operation procedure in User CRC/Checksum mode is described in the following list, and Figure 71 shows example program tip:

1. Select User CRC/Checksum Mode and Checksum.
2. Clear Flash CRC data register (FCDRH/FCDRL).
3. Read data from the flash memory.
4. Write the data to FCDIN Register.
5. Read the Checksum result.

Figure 71. Program Tip for Checksum Operation in User CRC/Checksum Mode

```
1          unsigned char code *rom_addr=0x0000;
2              unsigned int i=0;
3
4              FCCR |= _1000_0000;                          // Flash CRC User CRC/Checksum Mode
5              FCCR &= _0010_0000;                          // Flash CRC Checksum
6              FCCR |= _0100_0000;                          // Flash CRC data register clear
7
8              for(i=0x0000; i <= 0x1FFF; i++)      // 0000H~1FFFH
9              {
10                      FCDIN = rom_addr[i];
11                      WDTCR |= _0010_0000;                // Clear WDT counter
12             }
13
14         //**** Read Checksum result
15                 Temp0 = FCDRH;
16                 Temp1 = FCDRL;
```

17.3 Register Map

Table 28. Flash CRC/Checksum Generator Register Map

Name	Address	Direction	Default	Description
FCSARH	5050H (XSFR)	R/W	00H	Flash CRC Start Address High Register
FCEARH	5051H (XSFR)	R/W	00H	Flash CRC End Address High Register
FCSARM	5052H (XSFR)	R/W	00H	Flash CRC Start Address Middle Register
FCEARM	5053H (XSFR)	R/W	00H	Flash CRC End Address Middle Register
FCSARL	5054H (XSFR)	R/W	00H	Flash CRC Start Address Low Register
FCEARL	5055H (XSFR)	R/W	0FH	Flash CRC End Address Low Register
FCCR	5056H (XSFR)	R/W	00H	Flash CRC Control Register
FCDRH	5057H (XSFR)	R	FFH	Flash CRC Data High Register
FCDRL	5058H (XSFR)	R	FFH	Flash CRC Data Low Register
FCDIN	89H	R/W	00H	Flash CRC Data In Register

17.4 Register Description

FCSARH (Flash CRC Start Address High Register): 5050H (XSFR)

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	FCSARH0
–	–	–	–	–	–	–	R/W

Initial value: 00H

FCSARH0 Flash CRC Start Address High

NOTE:

1. Used only to Auto CRC Mode.

FCSARM (Flash CRC Start Address Middle Register): 5052H (XSFR)

7	6	5	4	3	2	1	0
FCSARM7	FCSARM6	FCSARM5	FCSARM4	FCSARM3	FCSARM2	FCSARM1	FCSARM0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

FCSARM[7:0] Flash CRC Start Address Middle

NOTE:

1. Used only to Auto CRC Mode.

FCSARL (Flash CRC Start Address Low Register): 5054H (XSFR)

7	6	5	4	3	2	1	0
FCSARL7	FCSARL6	FCSARL5	FCSARL4	FCSARL3	FCSARL2	FCSARL1	FCSARL0
R/W	R/W	R/W	R/W	–	–	–	–

Initial value: 00H

FCSARL[7:4] Flash CRC Start Address Low

NOTE:

1. Used only to Auto CRC Mode.

FCSARL[3:0] These bits are always “0000”.

FCEARH (Flash CRC End Address High Register): 5051H (XSFR)

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	FCEARH0
–	–	–	–	–	–	–	R/W

Initial value: 00H

FCEARH0 Flash CRC End Address High

NOTE:

1. Used only to Auto CRC Mode.

FCEARM (Flash CRC End Address Middle Register): 5053H (XSFR)

7	6	5	4	3	2	1	0
FCEARM7	FCEARM6	FCEARM5	FCEARM4	FCEARM3	FCEARM2	FCEARM1	FCEARM0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

FCEARM[7:0] Flash CRC End Address Middle

NOTE:

1. Used only to Auto CRC Mode.

FCEARL (Flash CRC End Address Low Register): 5055H (XSFR)

7	6	5	4	3	2	1	0
FCEARL7	FCEARL6	FCEARL5	FCEARL4	FCEARL3	FCEARL2	FCEARL1	FCEARL0
R/W	R/W	R/W	R/W	–	–	–	–

Initial value: 0FH

FCEARL[7:4] Flash CRC End Address Low

NOTE:

1. Used only to Auto CRC Mode.

FCEARL[3:0] These bits are always “1111”.

FCDRH (Flash CRC Data High Register): 5057H (XSFR)

7	6	5	4	3	2	1	0
FCDRH7	FCDRH6	FCDRH5	FCDRH4	FCDRH3	FCDRH2	FCDRH1	FCDRH0
R	R	R	R	R	R	R	R

Initial value: FFH

FCDRH[7:0] Flash CRC Data High

FCDRL (Flash CRC Data Low Register): 5058H (XSFR)

7	6	5	4	3	2	1	0
FCDRL7	FCDRL6	FCDRL5	FCDRL4	FCDRL3	FCDRL2	FCDRL1	FCDRL0
R	R	R	R	R	R	R	R

Initial value: FFH

FCDRL[7:0] Flash CRC Data Low

FCDIN (Flash CRC Data IN Register): 89H

7	6	5	4	3	2	1	0
FCDIN7	FCDIN6	FCDIN5	FCDIN4	FCDIN3	FCDIN2	FCDIN1	FCDIN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

FCDIN[7:0] Flash CRC Data In

NOTE:

1. Used only to User CRC Mode.

FCCR (Flash CRC Control Register): 5056H (XSFR)

7	6	5	4	3	2	1	0
CRCMOD	CDCL	MDSEL	–	CKSEL2	CKSEL1	CKSEL0	CRCRUN
R/W	R/W	R/W	–	R/W	R/W	R/W	R/W

Initial value: 00H

CRCMOD Select CRC/Checksum Mode

0 Auto CRC/Checksum Mode

1 User CRC/Checksum Mode

CDCL Flash CRC Data Register Clear

0 No effect

1 Clear Flash CRC Data register

NOTE:

1. This bit is cleared to '0' automatically, after Flash CRC Data register is cleared. The FCDRH/L is set to "FFH" if the MDSEL is set to '0' and "00H" if the MDSEL is set to '1'. Used only to User CRC/Checksum Mode.

MDSEL CRC/Checksum Selection

0 Select CRC

1 Select Checksum

CKSEL[2:0] Select Flash CRC/Checksum Clock

CKSEL2	CKSEL1	CKSEL0	Description
0	0	0	f_{HFIRC}
0	0	1	$f_{HFIRC}/2$
0	1	0	$f_{HFIRC}/4$
0	1	1	$f_{HFIRC}/8$
1	0	0	f_X(system clock)
Other values			Not used

CRCRUN CRC/Checksum Start Signal and Busy Flag, used only to Auto CRC/Checksum mode.

0 Indicates that CRC/Checksum operation is not running or has finished. When written '0', CRC/Checksum operation is finished by force even if CRC/Checksum operation is running. It has no effect to write '0' if CRC/Checksum is not running currently.

1 When written '1', CRC/Checksum operation starts, and this bit remains '1' if CRC/Checksum operation is on-going. This bit is cleared to '0' automatically after CRC/Checksum operation finishes.

NOTE:

1. If any of the setting f_{HFIRC}, $f_{HFIRC}/2$, $f_{HFIRC}/4$, and $f_{HFIRC}/8$ is selected, HFIRCE must be cleared to '0' at OSCCR.

18 Power Down Operation

A96L116 offers two power-down modes to minimize power consumption of itself. Programs under the two power saving modes IDLE and STOP, are stopped and power consumption is reduced considerably.

18.1 Peripheral Operation in IDLE/STOP Mode

Peripheral's operations during IDLE/STOP mode are described in Table 29.

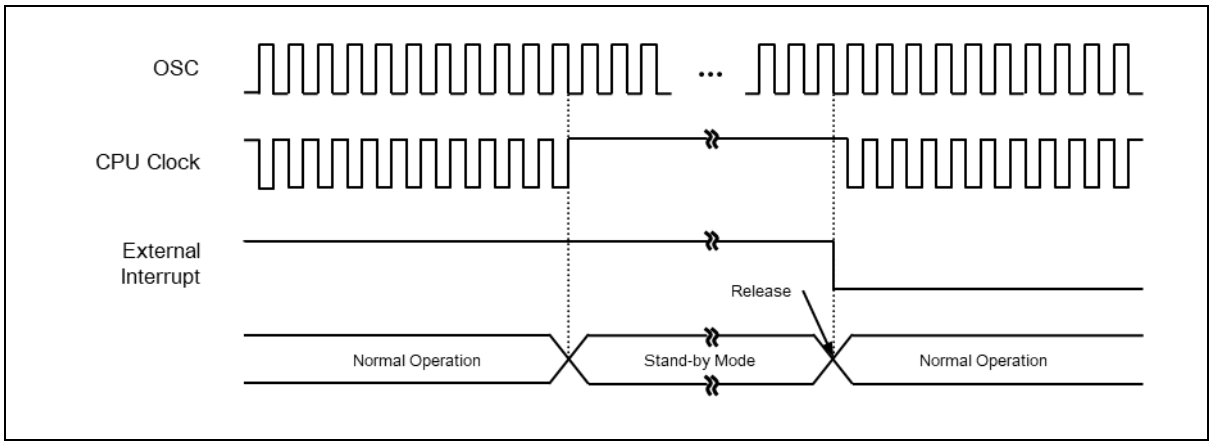
Table 29. Peripheral Operation during Power-down Mode

Peripheral	IDLE mode	STOP mode
CPU	All CPU operation is disable.	All CPU operations are disable
RAM	Retain	Retain
Basic Interval Timer	Operates continuously	Stop
Watchdog Timer	Operates continuously	Stop (Can be operated with WDTRC and LFIRC)
Timer 0	Operates continuously	Halted (only when the Event Counter Mode is Enabled or when f_{SUB}/f_{LFIRC} is selected for the clock of timer 0, Timer 0 operates Normally)
Timer 1 ~ 2	Operates continuously	Halted (only when the Event Counter Mode is Enabled, Timer operates Normally)
RTCC	Operates continuously	Can be operated with f_{SUB}
ADC	Operates continuously	Stop
LPUART	Operates continuously	Can be operated with f_{SUB}
I2C	Operates continuously	Only operate with external clock
SPI	Operates continuously	Only operate with external clock
HF-Internal OSC	Oscillation	Stop
LF-Internal OSC	Oscillation	Can be operated with setting value
WDTRC OSC (5 kHz)	Can be operated with setting value	Can be operated with setting value
I/O port	Retain	Retain
Control register	Retain	Retain
Address data bus	Retain	Retain
Release method	By RESET, all Interrupts	By RESET, Timer 0, Timer Interrupt (EC1, EC2), External Interrupt, WDT, LPUART, RTCC

18.2 IDLE Mode

The power control register is set to “01H” to enter the IDLE Mode. In this mode, the internal oscillation circuits remain active. Oscillation continues and peripherals are operated normally but CPU stops. It is released by reset or interrupt. To be released by interrupt, interrupt should be enabled before IDLE mode. If using reset, because the device becomes initialized state, the registers have reset value.

Figure 72. IDLE Mode Release Timing by External Interrupt



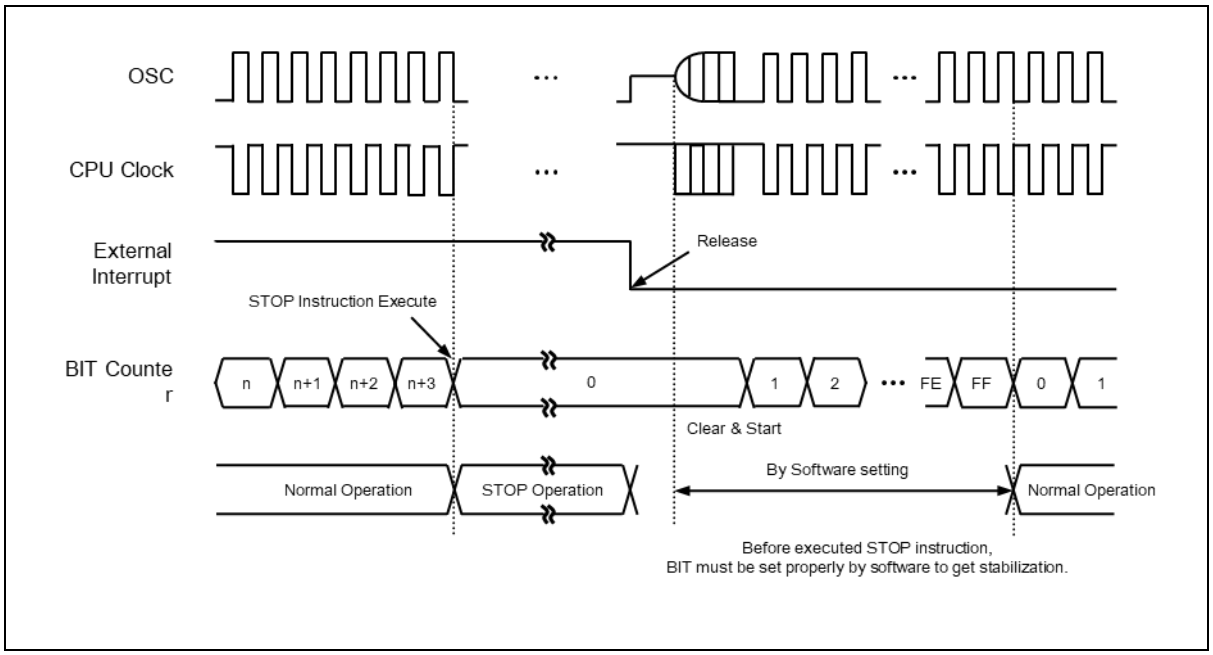
18.3 STOP Mode

The power control register is set to ‘03H’ to enter the STOP Mode. In the stop mode, the selected oscillator, system clock and peripheral clock are stopped. With the clock frozen, all functions are stopped, but the on-chip RAM and control registers are held.

The source for exit from STOP mode is hardware reset and interrupts. The reset re-defines all the control registers.

When exiting from STOP mode, enough oscillation stabilization time is required to normal operation. Figure 73 shows the timing diagram. When released from STOP mode, the Basic interval timer is activated on wake-up. Therefore, before STOP instruction, user must be set its relevant prescale divide ratio to have long enough time. This guarantees that the oscillator has started and stabilized.

Figure 73. STOP Mode Release Timing by External Interrupt

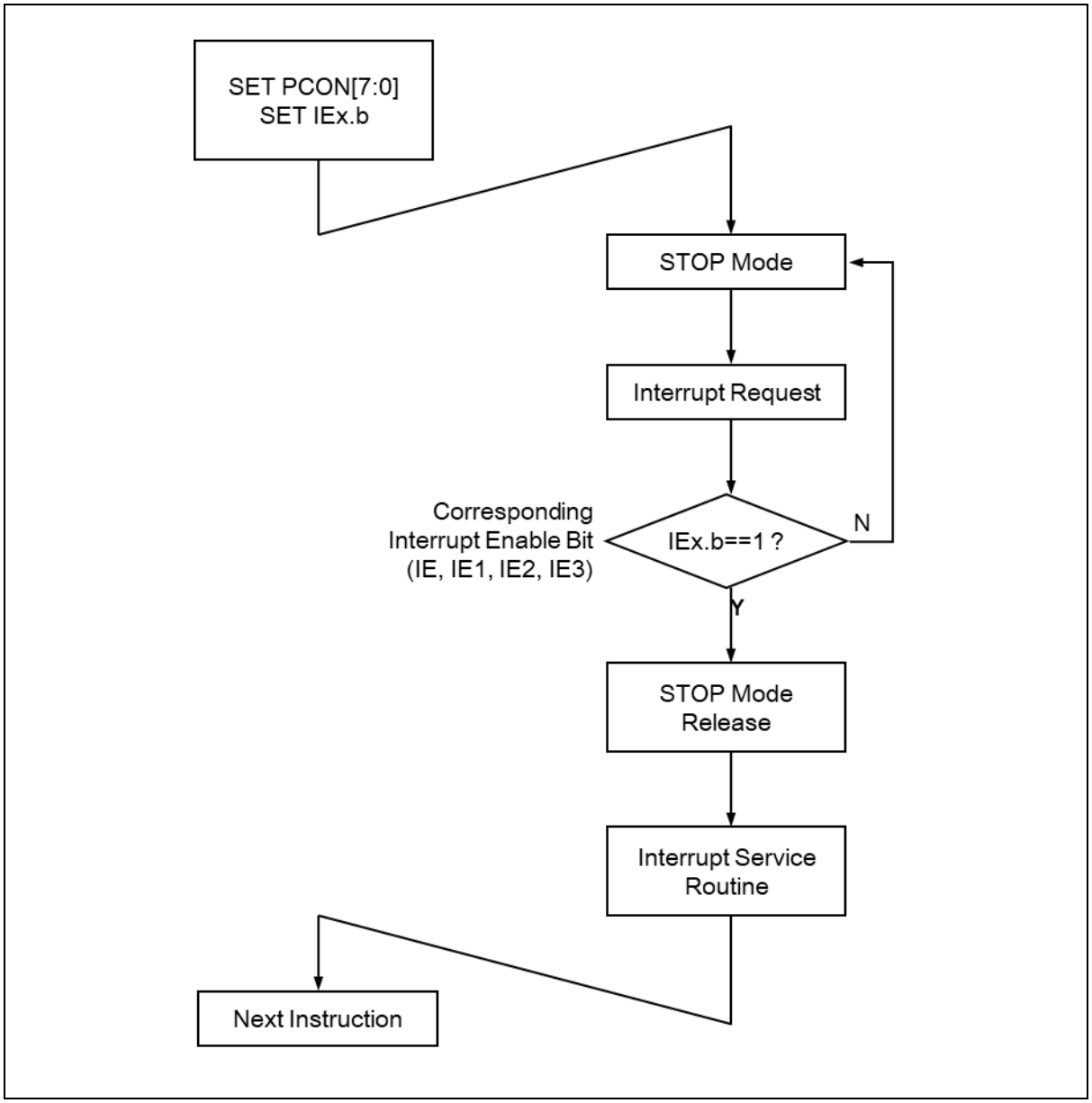


18.4 Release Operation of STOP Mode

After STOP mode is released, the operation begins according to content of related interrupt register just before STOP mode start (refer to Figure 74).

If the global interrupt Enable Flag (IE.EA) is set to ‘1’, the STOP mode is released by the interrupt which each interrupt enable flag = ‘1’ and the CPU jumps to the relevant interrupt service routine. Even if the IE.EA bit is cleared to ‘0’, the STOP mode is released by the interrupt of which the interrupt enable flag is set to ‘1’.

Figure 74. STOP Mode Release Flow



18.5 Register Map

Table 30. Power-down Operation Register Map

Name	Address	Direction	Default	Description
PCON	87H	R/W	00H	Power control register

18.6 Register Description

PCON (Power Control Register): 87H

7	6	5	4	3	2	1	0
–	–	–	–	–	–	PCON1	PCON0
–	–	–	–	–	–	R/W	R/W

Initial value: 00H

PCON[1:0]	Power Control
01H	IDLE mode enable
03H	STOP mode enable
Other Values	Normal operation

Before configuring a register PCON, please be aware of the followings:

To enter IDLE mode, PCON must be set to “01H”.

To enter STOP mode, PCON must be set to “03H”.

The PCON register is automatically cleared by a release signal in STOP/IDLE mode.

Three or more NOP instructions must follow immediately after the instruction that makes the device enter in STOP/IDLE mode. Refer to the following example code in Table 31.

Table 31. Example Code with 3 or More NOP Instructions

Example code 1

```
1
2  MOV     PCON, #01H      ; IDLE mode
3  NOP
4  NOP
5  NOP
6  •
7  •
8  •
```

Example code 2

```
1
2  MOV     PCON, #03H      ; STOP mode
3  NOP
4  NOP
5  NOP
6  •
7  •
8  •
```

19 Reset

When a reset event occurs, an internal register is selected to be initialized in accordance with a reset value. Each reset value described in Table 32 indicates a corresponding On-chip hardware that is to be initialized.

Table 32. Reset Value and the Relevant On-chip Hardware

On-Chip Hardware	Reset Value
Program Counter (PC)	0000H
Accumulator	00H
Stack Pointer (SP)	07H
Peripheral clock	On
Control register	Refer to peripheral registers.

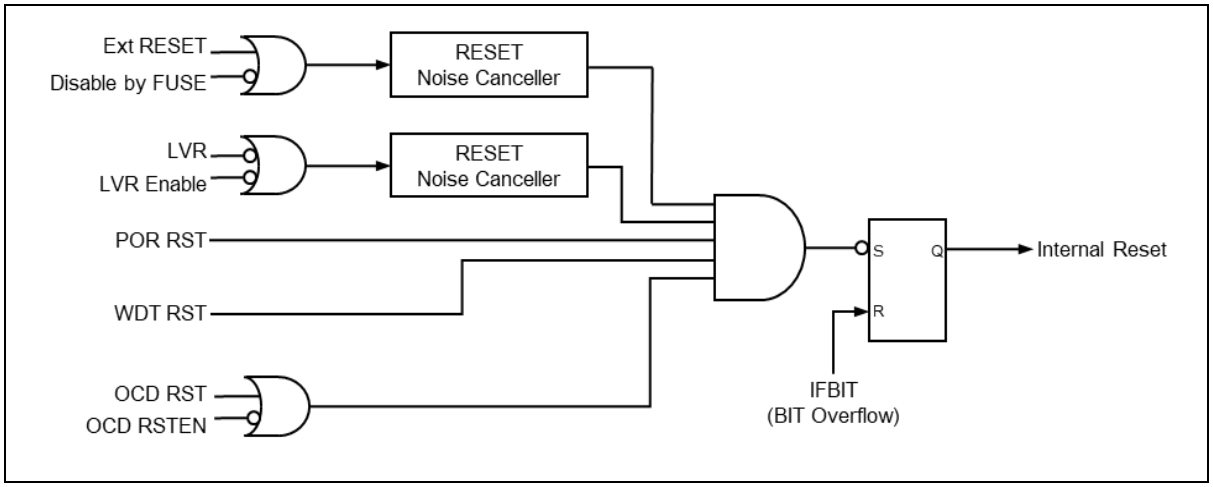
The A96L116 has five types of reset sources as listed in the followings:

- External RESETB
- Power-On RESET (POR)
- WDT Overflow Reset (in a case of WDTEN=1)
- Low-Voltage Reset (in a case of LVREN=0)
- OCD RESET

19.1 Reset Block Diagram

Figure 75 shows a reset block of A96L116.

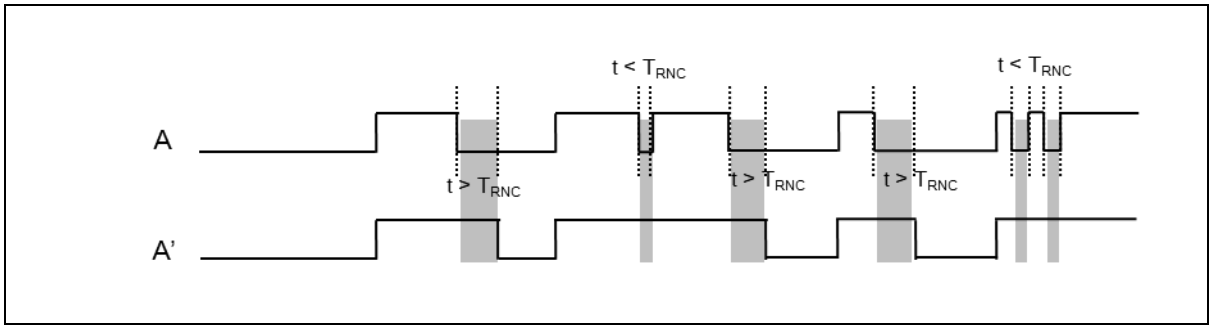
Figure 75. Reset Block Diagram



19.2 Reset Noise Canceller

Figure 76 is a noise canceller timing diagram for noise cancellation of RESET. It has the noise cancellation value of about 2 μs (@VDD=5 V) to the low input of system reset.

Figure 76. Reset Noise Canceller Timing Diagram



19.3 Power-on Reset

When device power is increasing, POR (Power-on Reset) executes a function to reset the device. If POR is used to reset the device, it executes the device reset function instead of RESET IC or RESET Circuit.

Figure 77. Fast VDD Rising Time

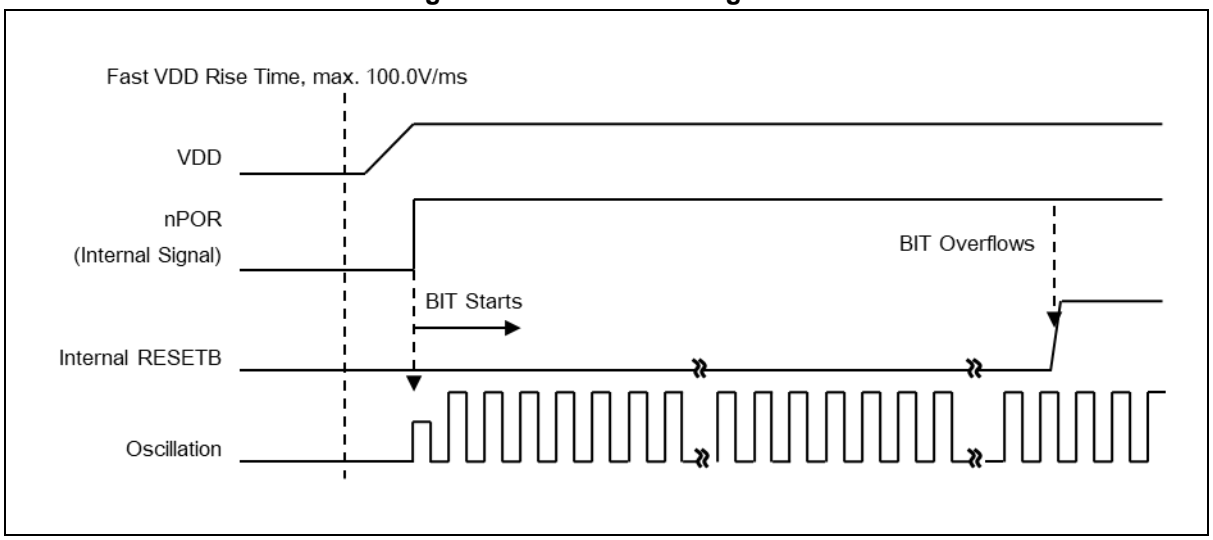


Figure 78. Internal Reset Release Timing on Power-Up

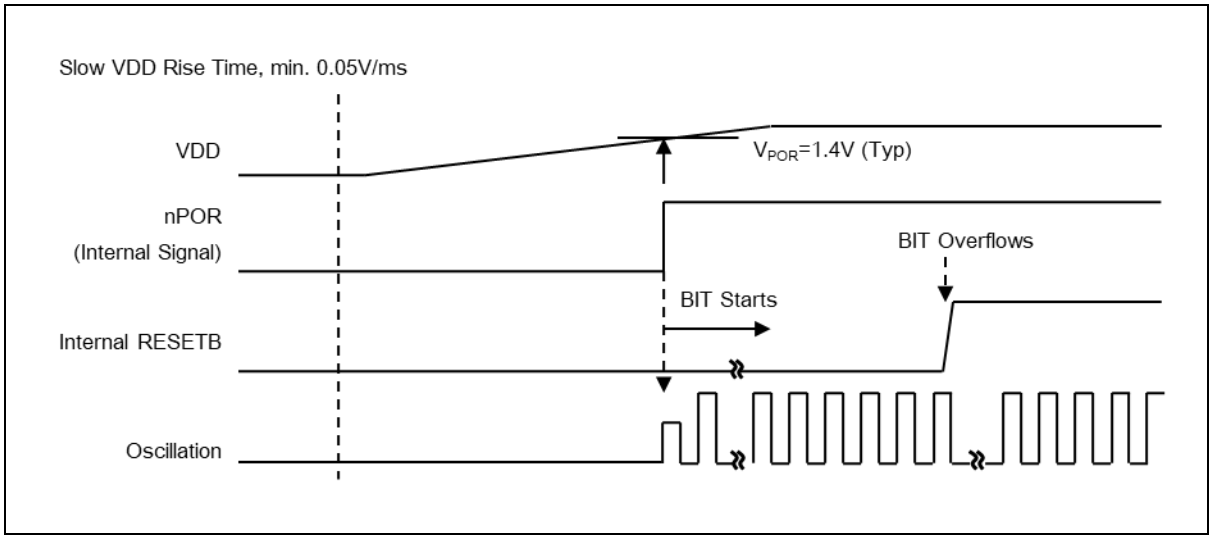
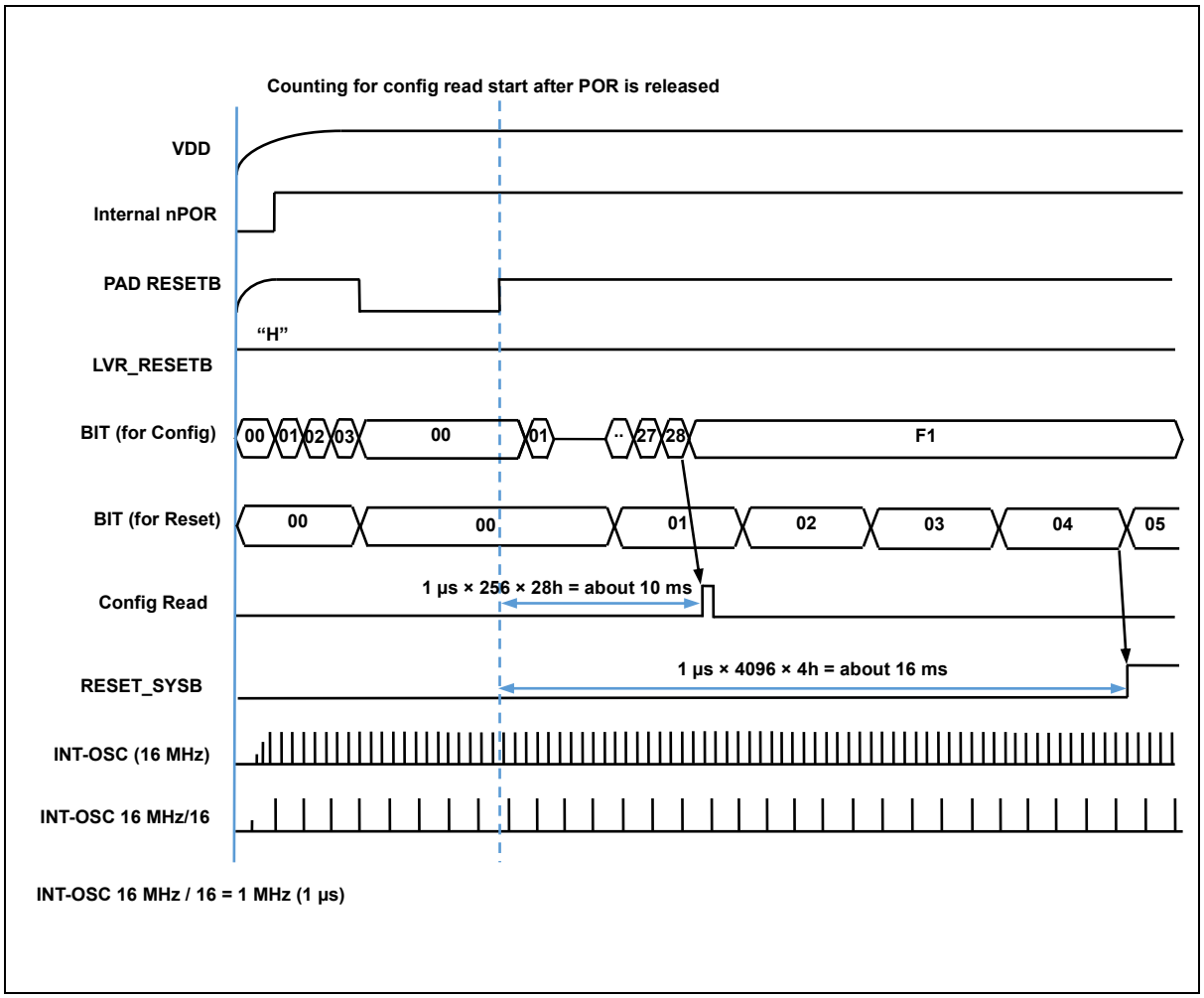


Figure 79. Configuration Timing When Powering On



Relationship between VDD input and internal oscillator is described in Figure 80 and Table 33.

Figure 80. Boot Process Waveform

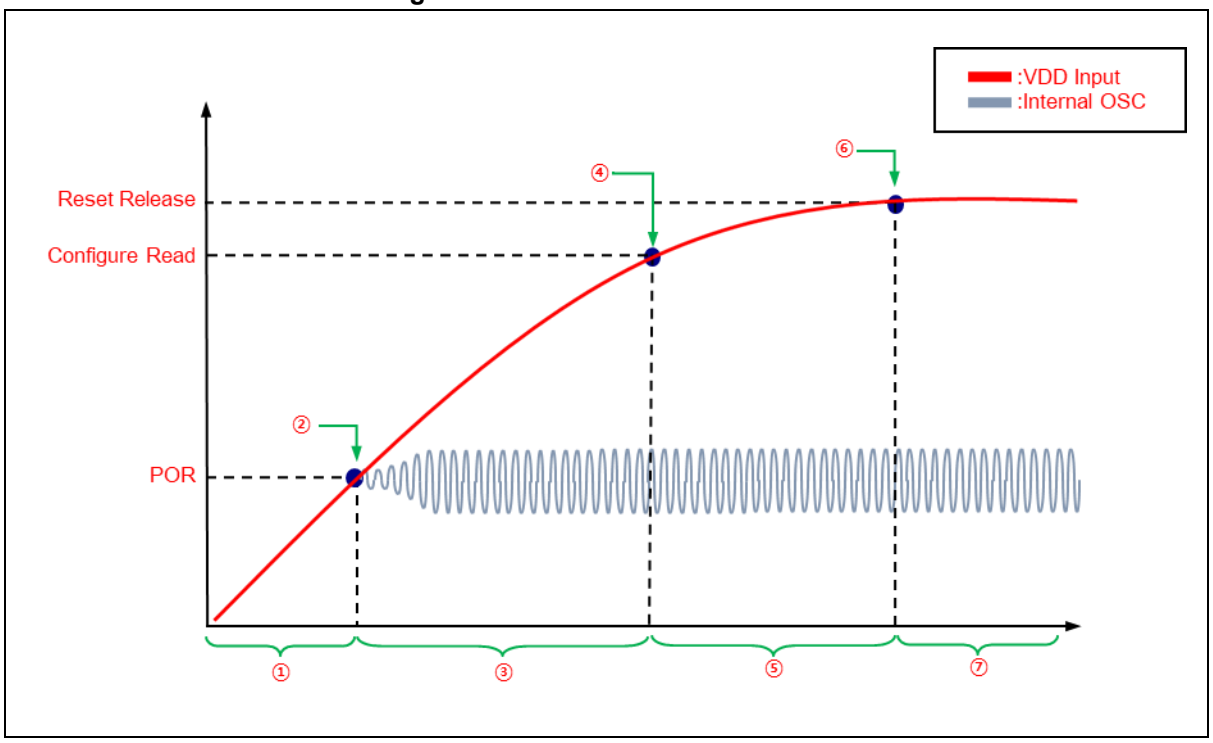


Table 33. Boot Process Description

Process	Description	Remark
①	No operation	
②	1st POR level detection	About 1.2 V
③	(INT-OSC 16 MHz/16) × 256 × 28h Delay section (=10 ms) VDD input voltage must rise over than flash operating voltage for Config read.	Slew rate ≥ 0.05 V/ms
④	Config read point	About 1.5 V ~ 1.6 V Config Value is determined by Writing Option.
⑤	Rising section to reset release level	16ms point after POR or Ext_reset release
⑥	Reset release section (BIT overflow) After 16 ms, after external reset release (external reset) 16 ms point after POR (POR only)	BIT is used for peripheral stability.
⑦	Normal operation	

19.4 External RESETB Input

External RESETB is the input to a Schmitt-trigger. If the RESETB pin is held with low for at least 10 μs over within the operating voltage range and stable oscillation, it is applied, and the internal state is initialized. When reset state becomes '1', it needs stabilization time with 16 ms and after the stable state, the internal RESET becomes '1'. The reset process step needs five oscillator clocks. And the program execution starts at the vector address stored at address 0000H.

Figure 81. Timing Diagram after RESET

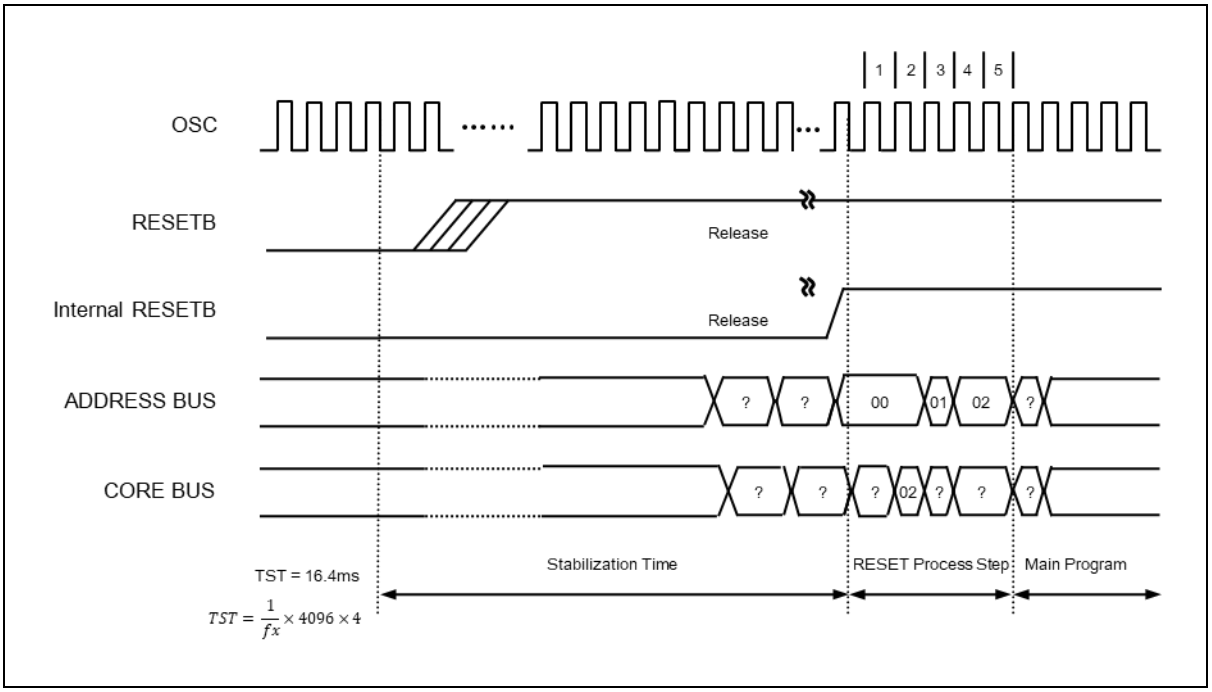
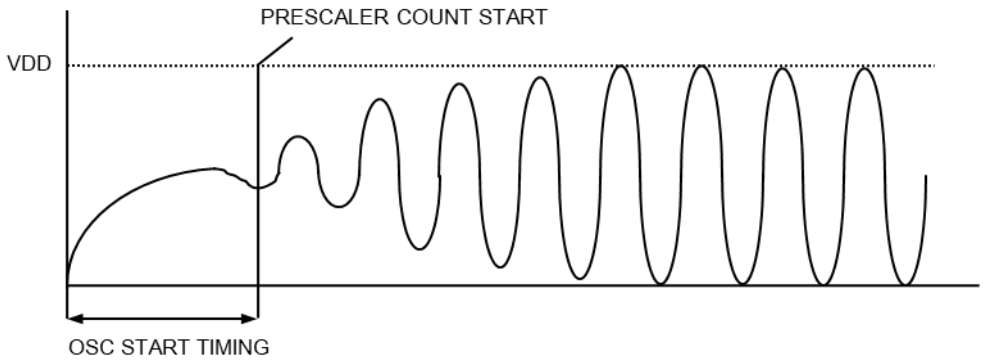


Figure 82. Oscillator Generating Waveform Example



NOTES:

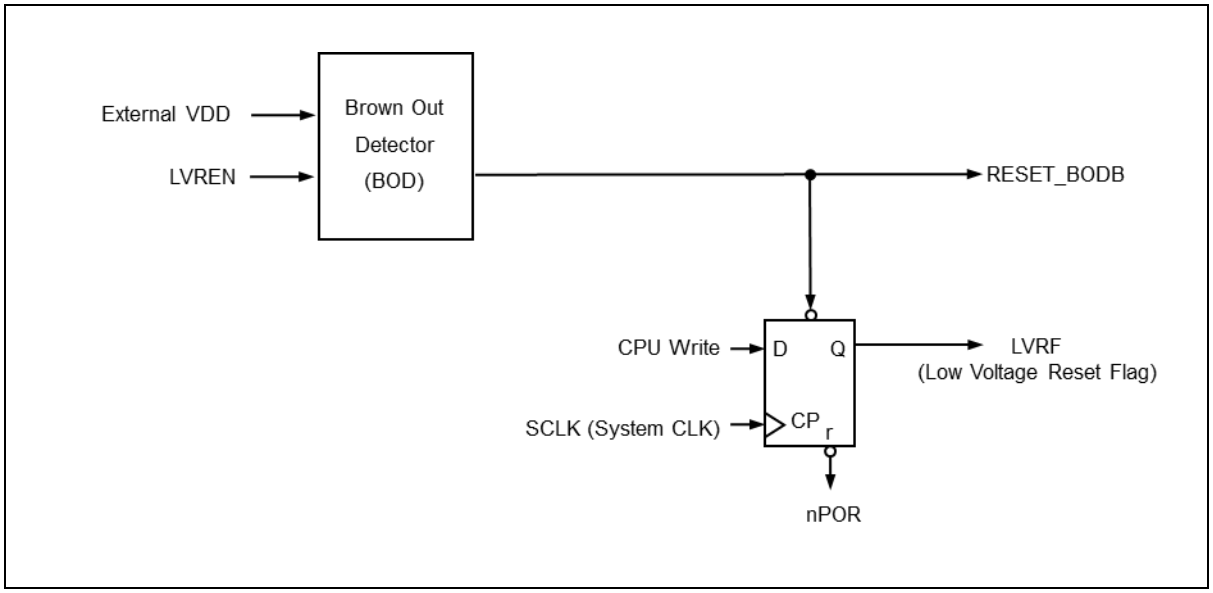
1. Stable generating time is not included in the start-up time.
2. RESETB pin has a pull-up resistor by hardware.

19.5 Brown-out Detector Processor

The A96L116 has an On-chip brown-out detection circuit (BOD) to monitor VDD level during its operation. It compares VDD level to a fixed trigger level which can be selected to be one of 1.50 V, 1.87 V, 2.02 V, 2.17 V, 2.32 V, 2.47 V, 2.64 V, 2.78 V by the LVRVS[2:0] bits. In STOP mode, since the BOD will contribute significantly to the total current consumption, the LVREN bit is set to off by software to minimize the current consumption.

19.5.1 Block Diagram

Figure 83. BOD Block Diagram



19.5.2 Internal Reset and BOD Reset in Timing Diagram

Figure 84. Internal Reset at Power Fail Situation

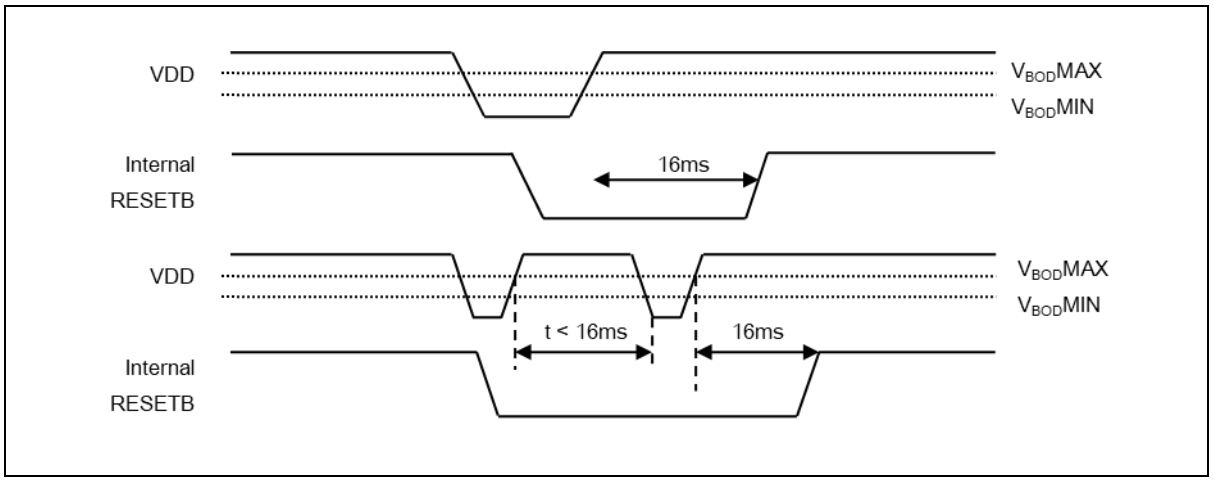


Figure 85. Configuration Timing when BOD Reset

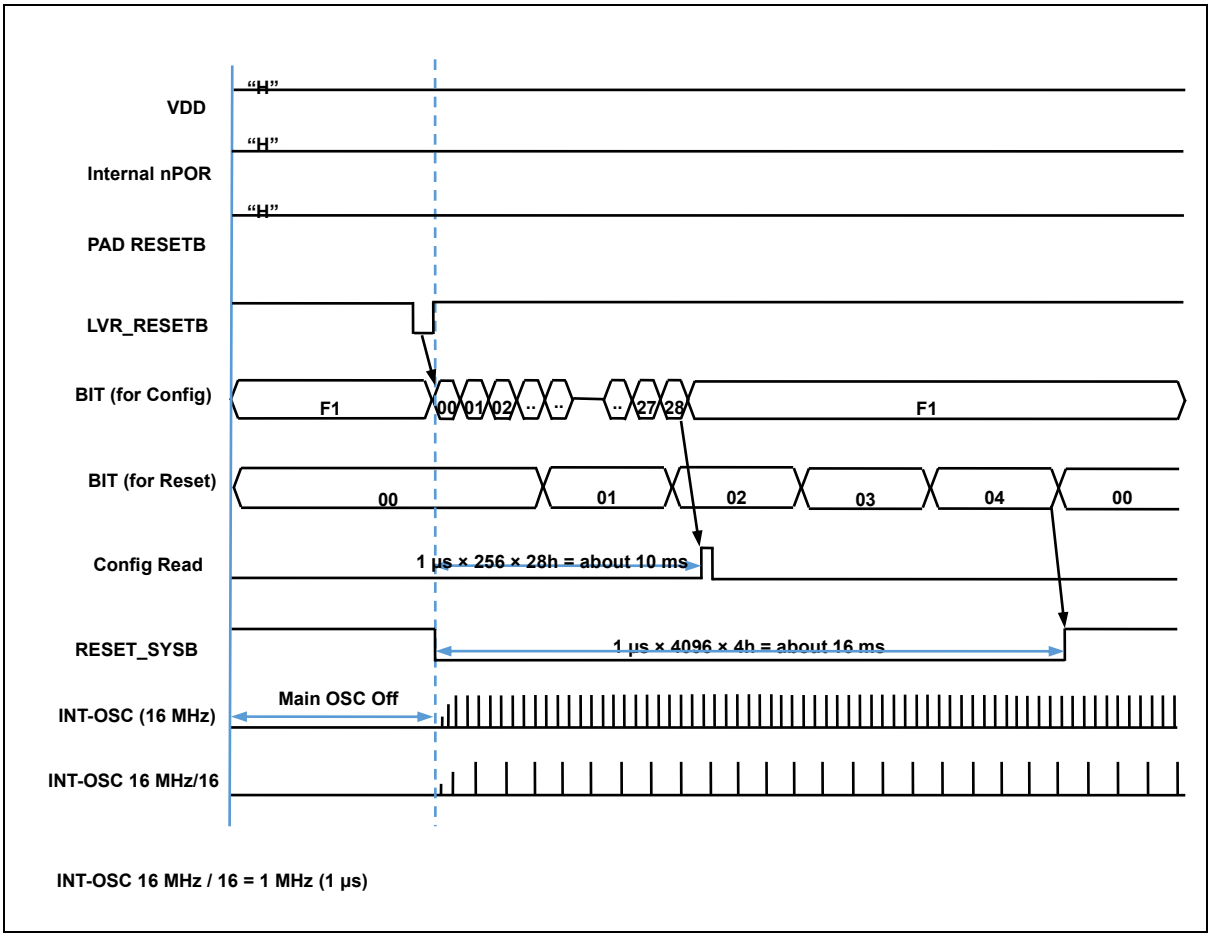
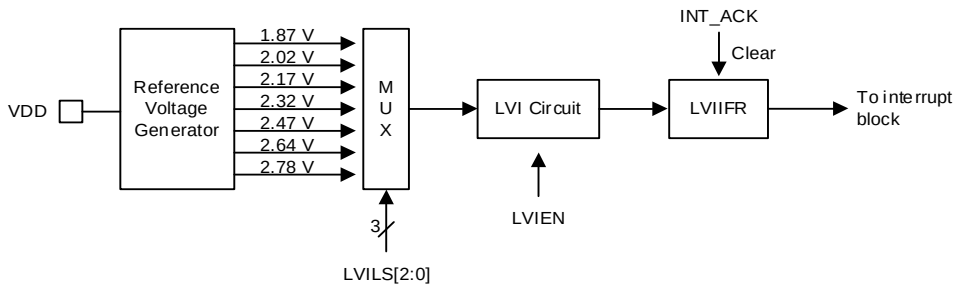


Figure 86. LVI Block Diagram



NOTES:

1. The LVIIFR bit will be set to '1' if the VDD goes down lower than a selected LVI level during LVIEN='1'.
2. The LVIIFR bit won't be set to '1' if the bit is cleared to '0' while the VDD remains the lower voltage than the LVI level.
3. The LVIIFR bit may be set to '1' if the LVIEN bit is set to '1' again after the LVIEN bit is cleared to '0' during VDD < the selected LVI level.

19.6 Register Map

Table 34. Reset Operation Register Map

Name	Address	Direction	Default	Description
RSTFR	E8H	R/W	80H	Reset Flag Register
LVRCR	D8H	R/W	00H	Low-Voltage Reset Control Register
LVRIDR	505FH (XSFR)	R/W	00H	LVR Write Identification Register
LVICR	86H	R/W	00H	Low-Voltage Indicator Control Register

19.7 Register Description

RSTFR (Reset Flag Register): E8H

7	6	5	4	3	2	1	0
PORF	EXTRF	WDTRF	OCDRF	LVRF	–	–	–
R/W	R/W	R/W	R/W	R/W	–	–	–

Initial value: 80H

PORF Power-On Reset flag bit. The bit is reset by writing '0' to this bit.
- 0 No detection
- 1 Detection

EXTRF External Reset (RESETB) flag bit. The bit is reset by writing '0' to this bit or by Power-On Reset.
- 0 No detection
- 1 Detection

WDTRF Watchdog Reset flag bit. The bit is reset by writing '0' to this bit or by Power-On Reset.
- 0 No detection
- 1 Detection

OCDRF On-Chip Debug Reset flag bit. The bit is reset by writing '0' to this bit or by Power-On Reset.
- 0 No detection
- 1 Detection

LVRF Low-Voltage Reset flag bit. The bit is reset by writing '0' to this bit or by Power-On Reset.
- 0 No detection
- 1 Detection

NOTES:

1. When the Power-On Reset occurs, the PORF bit is only set to '1', the WDTRF/OCDRF bits are cleared to "0".
2. When the Power-On Reset occurs, the EXTRF bit is unknown, at that time, the EXTRF bit can be set to '1' when External Reset (RESETB) occurs.

   When the Power-On Reset occurs, the LVRF bit is unknown, at that time, the LVRF bit can be set to '1' when LVR Reset occurs.
3. When a reset except the POR occurs, the corresponding flag bit is only set to '1', the other flag bits are kept in the previous values.

LVRCR (Low-Voltage Reset Control Register): D8H

7	6	5	4	3	2	1	0
LVRST	–	–	–	LVRVS2	LVRVS1	LVRVS0	LVREN
R/W	–	–	–	R/W	R/W	R/W	R/W

Initial value: 00H

LVRST — LVR Enable when Stop Release

0	Not effect at stop release
1	LVR enable at stop release

NOTES:

1. When this bit is '1', the LVREN bit is cleared to '0' by stop mode release. (LVR enable)
2. When this bit is '0', the LVREN bit is not affected by stop mode release.

LVRVS[2:0] — LVR Voltage Select

LVRVS2	LVRVS1	LVRVS0	Description
0	0	0	1.50 V
0	0	1	1.87 V
0	1	0	2.02 V
0	1	1	2.17 V
1	0	0	2.32 V
1	0	1	2.47 V
1	1	0	2.64 V
1	1	1	2.78 V

LVREN — LVR Operation

0	LVR Enable
1	LVR Disable

NOTES:

1. The LVRST and LVRVS[1:0] bits are cleared by a Power-On Reset but are retained by other reset signals.
2. The LVRVS[1:0] bits should be set to "00" while LVREN bit is '1'.
3. This register can be written with valid ID value (LVRIDR = 0x59).

LVRIDR (LVR Write Identification Register): 505FH (XSFR)

7	6	5	4	3	2	1	0
LVRID7	LVRID6	LVRID5	LVRID4	LVRID3	LVRID2	LVRID1	LVRID0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

LVRID[7:0] — LVR Write Identification

Others	No identification value
01011001	Identification value for LVR register write

(These bits are automatically cleared to logic "00H" immediately after one time operation)

LVICR (Low-Voltage Indicator Control Register): 86H

7	6	5	4	3	2	1	0
–	–	LVIIFR	LVIEN	–	LVILS2	LVILS1	LVILS0
–	–	R/W	R/W	–	R/W	R/W	R/W

Initial value: 00H

LVIIFR — When Low-Voltage Indicator Interrupt occurs, this bit becomes '1'. For clearing bit, write '0' to this bit or auto clear by INT_ACK signal.

0	No detection
1	Detection

LVIEN — LVI Enable/disable

0	Disable
1	Enable

LVILS[2:0] — LVI Level Select

LVILS2	LVILS1	LVILS0	Description
0	0	0	1.87 V
0	0	1	1.87 V
0	1	0	2.02 V
0	1	1	2.17 V
1	0	0	2.32 V
1	0	1	2.47 V
1	1	0	2.64 V
1	1	1	2.78 V

20 Flash Memory

The A96L116 incorporates flash memory inside. The program can be written, erased, and overwritten on flash memory while mounted on a board. The flash memory can be read by ‘MOVC’ instruction and programmed in OCD, serial ISP, or user program mode.

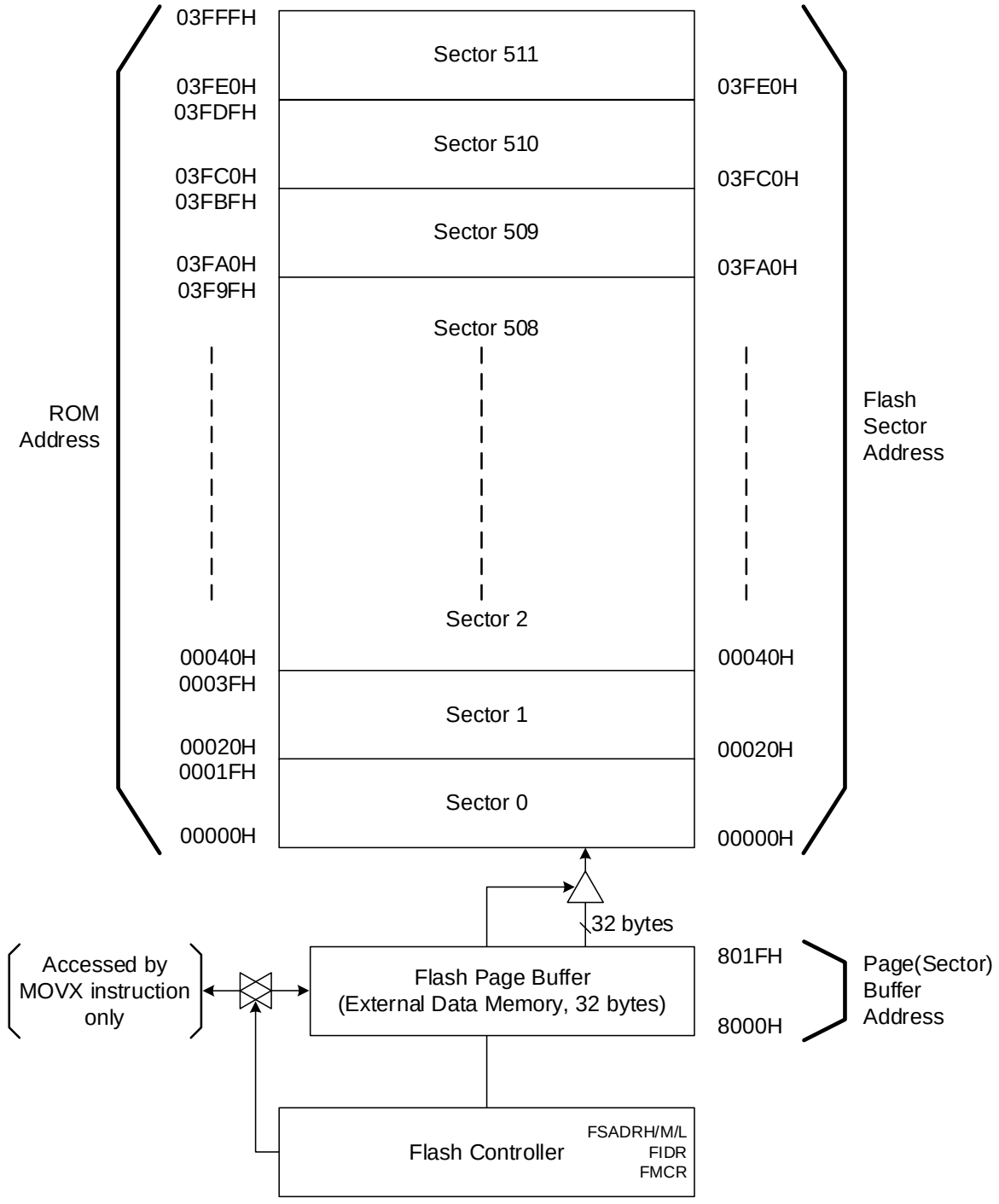
- Flash memory size: 16 Kbytes
- Single power supply program and erase
- Command interface for fast program and erase operation
- Up to 10,000 program/erase cycles at typical voltage and temperature for flash memory

NOTE:

1. The RXE0 bit of LPUT0CR1 register should be disabled before flash memory erase and write start.

20.1 Flash Program ROM Structure

Figure 87. Flash Program ROM Structure



NOTES:

1. The flash memory doesn't support byte erase mode.
2. The flash shouldn't be entered in erase/write mode during Data Flash program. So before the flash program start, the Data Flash busy bit (DFMBUSY) should be checked whether the bit is '1' or not. If the bit is '1', do not start flash program.

20.2 Register Map

Table 35. Flash Memory Register Map

Name	Address	Direction	Default	Description
FSADRH	FAH	R/W	00H	Flash Sector Address High Register
FSADRM	FBH	R/W	00H	Flash Sector Address Middle Register
FSADRL	FCH	R/W	00H	Flash Sector Address Low Register
FIDR	FDH	R/W	00H	Flash Identification Register
FMCR	FEH	R/W	00H	Flash Mode Control Register

20.3 Register Description

FSADRH (Flash Sector Address High Register): FAH

7	6	5	4	3	2	1	0
–	–	–	–	FSADRH3	FSADRH2	FSADRH1	FSADRH0
–	–	–	–	R/W	R/W	R/W	R/W

Initial value: 00H

FSADRH[3:0] Flash Sector Address High

FSADRM (Flash Sector Address Middle Register): FBH

7	6	5	4	3	2	1	0
FSADRM7	FSADRM6	FSADRM5	FSADRM4	FSADRM3	FSADRM2	FSADRM1	FSADRM0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

FSADRM[7:0] Flash Sector Address Middle

FSADRL (Flash Sector Address Low Register): FCH

7	6	5	4	3	2	1	0
FSADRL7	FSADRL6	FSADRL5	FSADRL4	FSADRL3	FSADRL2	FSADRL1	FSADRL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

FSADRL[7:0] Flash Sector Address Low

FIDR (Flash Identification Register): FDH

7	6	5	4	3	2	1	0
FIDR7	FIDR6	FIDR5	FIDR4	FIDR3	FIDR2	FIDR1	FIDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

FIDR[7:0] Flash Identification

Others	No identification value
10100101	Identification value for the flash mode

(These bits are automatically cleared to logic “00H” immediately after one time operation except “Flash page buffer reset mode”)

FMCR (Flash Mode Control Register): FEH

7	6	5	4	3	2	1	0
FMBUSY	–	–	–	–	FMCR2	FMCR1	FMCR0
R	–	–	–	–	R/W	R/W	R/W

Initial value: 00H

FMBUSY Flash Mode Busy Bit. This bit will be used for only debugger.

0	No effect when ‘1’ is written
1	Busy

FMCR[2:0] Flash Mode Control Bits. During a Flash mode operation, the CPU is hold and the global interrupt is on disable-state regardless of the IE.7 (EA) bit.

FMCR2	FMCR1	FMCR0	Description
0	0	1	Select Flash page buffer reset mode and start regardless of the FIDR value (clear all 32 bytes to ‘0’).
0	1	0	Select Flash sector erase mode and start operation when the FIDR[7:0]=”10100101”
0	1	1	Select Flash sector write mode and start operation when the FIDR[7:0]=”10100101”
1	0	0	Select Flash hard lock and start operation when the FIDR[7:0]=”10100101”

Other Values: No operation

(These bits are automatically cleared to logic “00H” immediately after one time operation)

20.4 Serial In-System Program (ISP) Mode

Serial in-system program uses the interface of debugger which uses two wires. Refer to the development tool chapter in the A96L116 datasheet for details about the debugger.

20.5 Protection Area (User Program Mode)

A user can program flash memory (protection area) of the A96L116. The protection area cannot be erased or programmed if any protection area is enabled by the configure option 2. If the protection area is disabled (PAEN=0), this area can be erased or programmed.

The user can choose the size of protection area by using configure option 2. For more information about configure option 2, please refer to Appendix A. Configure option.

Table 36 describes protection area size and relative information.

Table 36. Protection Area Size and Relative Information

Protection Area Size Select			Size of Protection Area	Address of Protection Area
PASS2	PASS1	PASS0		
0	0	0	0.7 Kbytes	0100H – 03FFH
0	0	1	1.7 Kbytes	0100H – 07FFH
0	1	0	2.7 Kbytes	0100H – 0BFFH
0	1	1	3.8 Kbytes	0100H – 0FFFH
1	0	0	13.7 Kbytes	0100H – 37FFH
1	0	1	14.7 Kbytes	0100H – 3BFFH
1	1	0	15.2 Kbytes	0100H – 3DFFH
1	1	1	15.5 Kbytes	0100H – 3EFFH

NOTE:

1. Refer to Appendix A. Configure option.

20.6 Erase Mode

The sector erase program procedure in user program mode:

1. Page buffer clear (FMCR=0x01)
2. Write ‘0’ to the page buffer.
3. Set Flash sector address register (FSADRH/FSADRM/FSADRL).
4. Set Flash identification register (FIDR).
5. Check User ID to prevent invalid operation [NOTE].
6. Set Flash mode control register (FMCR).
7. Erase verify.

NOTES:

1. Refer to a subsection 20.8 Protection for invalid erase/write.
2. The RXE0 bit of LPUT0CR1 register should be disabled before flash memory erase and write start.

Figure 88 shows example program tip regarding sector erase.

Figure 88. Program Tip: Sector Erase

```
1             MOV      FMCR,#0x01        ;page buffer clear
2             NOP                        ;Dummy instruction, This instruction must be needed.
3             NOP                        ;Dummy instruction, This instruction must be needed.
4             NOP                        ;Dummy instruction, This instruction must be needed.
5
6             MOV      A,#0
7             MOV      R0,#SectorSize    ;Sector size of Device
8             MOV      DPH,#0x80         ;Page Buffer Address is 8000H
9             MOV      DPL,#0
10
11   Pgbuf_clr:
12            MOVX     @DPTR,A
13            INC      DPTR
14            DJNZ     R0,Pgbuf_clr      ;Write '0' to all page buffer
15
16
17            MOV      FSADRH,#SAH       ;Sector Address High Byte.
18            MOV      FSADRM,#SAM       ;Sector Address Middle Byte
19            MOV      FSADRL,#SAL       ;Sector Address Low Byte
20            MOV      FIDR,#0xA5        ;Identification value
21
22            MOV      A,#ID_DATA_1      ;Check the UserID(written by user)
23            CJNE     A,UserID1,No_WriteErase     ;This routine for UserID must be needed.
24            MOV      A,#ID_DATA_2
25            CJNE     A,UserID2,No_WriteErase
26
27            MOV      FMCR,#0x02        ;Start Flash erase mode
28            NOP                        ;Dummy instruction, This instruction must be needed.
29            NOP                        ;Dummy instruction, This instruction must be needed.
30            NOP                        ;Dummy instruction, This instruction must be needed.
31
32            LJMP     Erase_verify
33                     ---
34   No_WriteErase:
35            MOV      FIDR,#00H
36            MOV      UserID1,#00H
37            MOV      UserID2,#00H
38                     ---
39   Erase_verify:
40                     ---
41   Verify_error:
42                     ---
```

20.7 Write Mode

The sector Write program procedure in user program mode:

1. Page buffer clear (FMCR=0x01)
2. Write data to page buffer.
3. Set Flash sector address register (FSADRH/FSADRM/FSADRL).
4. Set Flash identification register (FIDR).
5. Check the UserID to prevent the invalid operation **NOTE1**.
6. Set Flash mode control register (FMCR).
7. Write verify.

NOTES:

1. Refer to "20.8. Protection for Invalid Erase/Write".
2. All data of the sector should be "00H" before writing data to a sector.
3. The RXE0 bit of LPUT0CR1 register should be disabled before flash memory erase and write start.

Figure 89 shows example program tip regarding sector write.

Figure 89. Program Tip: Sector Write

```
1             MOV      FMCR,#0x01        ;page buffer clear
2             NOP                        ;Dummy instruction, This instruction must be needed.
3             NOP                        ;Dummy instruction, This instruction must be needed.
4             NOP                        ;Dummy instruction, This instruction must be needed.
5
6             MOV      A,#0
7             MOV      R0,#SectorSize    ;Sector size of Device
8             MOV      DPH,#0x80         ;Page Buffer Address is 8000H
9             MOV      DPL,#0
10
11   Pgbuf_WR:MOVX     @DPTR,A
12            INC      A
13            INC      DPTR
14            DJNZ     R0,Pgbuf_WR       ;Write data to all page buffer
15
16            MOV      FSADRH,#SAH       ;Sector Address High Byte.
17            MOV      FSADRM,#SAM       ;Sector Address Middle Byte
18            MOV      FSADRL,#SAL       ;Sector Address Low Byte
19            MOV      FIDR,#0xA5        ;Identification value
20
21            MOV      A,#ID_DATA_1      ;Check the UserID(written by user)
22            CJNE     A,UserID1,No_WriteErase     ;This routine for UserID must be needed.
23            MOV      A,#ID_DATA_2
24            CJNE     A,UserID2,No_WriteErase
25
26            MOV      FMCR,#0x03        ;Start Flash write mode
27            NOP                        ;Dummy instruction, This instruction must be needed.
28            NOP                        ;Dummy instruction, This instruction must be needed.
29            NOP                        ;Dummy instruction, This instruction must be needed.
30
31            LJMP     Write_verify
32                     ---
33   No_WriteErase:
34            MOV      FIDR,#00H
35            MOV      UserID1,#00H
36            MOV      UserID2,#00H
37                     ---
38   Write_verify:
39                     ---
40   Verify_error:
41                     ---
```

The Byte Write program procedure in user program mode:

1. Page buffer clear (FMCR=0x01)
2. Write data to page buffer.
3. Set Flash sector address register (FSADRH/FSADRM/FSADRL).
4. Set Flash identification register (FIDR).
5. Check the UserID to prevent the invalid operation [NOTE1].
6. Set Flash mode control register (FMCR).
7. Write verify.

NOTES:

1. Refer to "20.8. Protection for invalid erase/write".
2. The data of the address should be "00H" before writing data to an address.
3. The RXE0 bit of LPUT0CR1 register should be disabled before flash memory erase and write start.

Figure 90 shows example program tip regarding byte write.

Figure 90. Program Tip: Byte Write

```
1             MOV      FMCR,#0x01        ;page buffer clear
2             NOP                        ;Dummy instruction, This instruction must be needed.
3             NOP                        ;Dummy instruction, This instruction must be needed.
4             NOP                        ;Dummy instruction, This instruction must be needed.
5
6             MOV      A,#5
7             MOV      DPH,#0x80
8             MOV      DPL,#0
9             MOVX     @DPTR,A           ;Write data to page buffer
10
11            MOV      A,#6
12            MOV      DPH,#0x80
13            MOV      DPL,#0x05
14            MOVX     @DPTR,A           ;Write data to page buffer
15
16            MOV      FSADRH,#SAH       ;Sector Address High Byte.
17            MOV      FSADRM,#SAM       ;Sector Address Middle Byte
18            MOV      FSADRL,#SAL       ;Sector Address Low Byte
19            MOV      FIDR,#0xA5        ;Identification value
20
21            MOV      A,#ID_DATA_1      ;Check the UserID(written by user)
22            CJNE     A,UserID1,No_WriteErase     ;This routine for UserID must be needed.
23            MOV      A,#ID_DATA_2
24            CJNE     A,UserID2,No_WriteErase
25
26            MOV      FMCR,#0x03        ;Start Flash write mode
27            NOP                        ;Dummy instruction, This instruction must be needed.
28            NOP                        ;Dummy instruction, This instruction must be needed.
29            NOP                        ;Dummy instruction, This instruction must be needed.
30
31            LJMP     Write_verify
32                     ---
33    No_WriteErase:
34            MOV      FIDR,#00H
35            MOV      UserID1,#00H
36            MOV      UserID2,#00H
37                     ---
38    Write_verify:
39                     ---
40    Verify_error:
41                     ---
```

20.8 Protection for Invalid Erase/Write

It needs to be careful when programming Flash erase/write operation in code. In addition, it needs preparations for invalid jump to the Flash erase/write code occurred by malfunction, noise, and power off.

NOTE:

1. For more information about the invalid erase and write operation, please refer to Appendix C. Flash Protection for Invalid Erase/Write.

Follow the procedure below to protect for invalid erase and write operations:

1. User ID check routine for the Flash erase/write code.

Figure 91. User ID Check Routine for Flash Erase/Write Code

```
ErWt_rtn:
                    ---
            MOV     FIDR,#10100101B             ;ID Code
            MOV     A,#ID_DATA_1                ;Ex) ID_DATA_1: 93H, ID_DATA_2: 85H, ID_DATA_3: 5AH
            CJNE    A,UserID1,No_WriteErase
            MOV     A,#ID_DATA_2
            CJNE    A,UserID2,No_WriteErase
            MOV     A,#ID_DATA_3
            CJNE    A,UserID3,No_WriteErase
            MOV     FMCR,#0x??                  ;0x03 if write, 0x02 if erase
                    ---
                    ---
            RET

No_WriteErase:
            MOV     FIDR,#00H
            MOV     UserID1,#00H
            MOV     UserID2,#00H
            MOV     UserID3,#00H
            MOV     Flash_flag,#00H
            RET
```

NOTES:

1. Use the code in Figure 91 to avoid invalid Flash erase/write.
2. It is important that the location where the UserID1/2/3 will be written. The invalid Flash erase/write problem will remain if the UserID1/2/3 is written at the above line of the instruction “MOV FIDR,#10100101B”. Therefore, it is recommended to write the UserID1/2/3 in a different routine after returning.

Figure 92 shows example code regarding the recommendation.

Figure 92. Example Code Regarding the Recommendation

```
Decide_ErWt:
            ---
            MOV     Flash_flag1,#38H  ;Random value for example, in case of erase/write needs
            MOV     FSADRL,#20H       ;Here 20H is example,
            MOV     Flash_flag2,#75H
            RET
```

2. The Flash sector address (FSADRH/FSADRM/FSADRL) must always keep the address of the flash memory used for data area. For example, The FSADRH and FSADRM values must be set to “0x00” and “0x0f” if the range from 0x0f00 to 0x0fff is used for data.

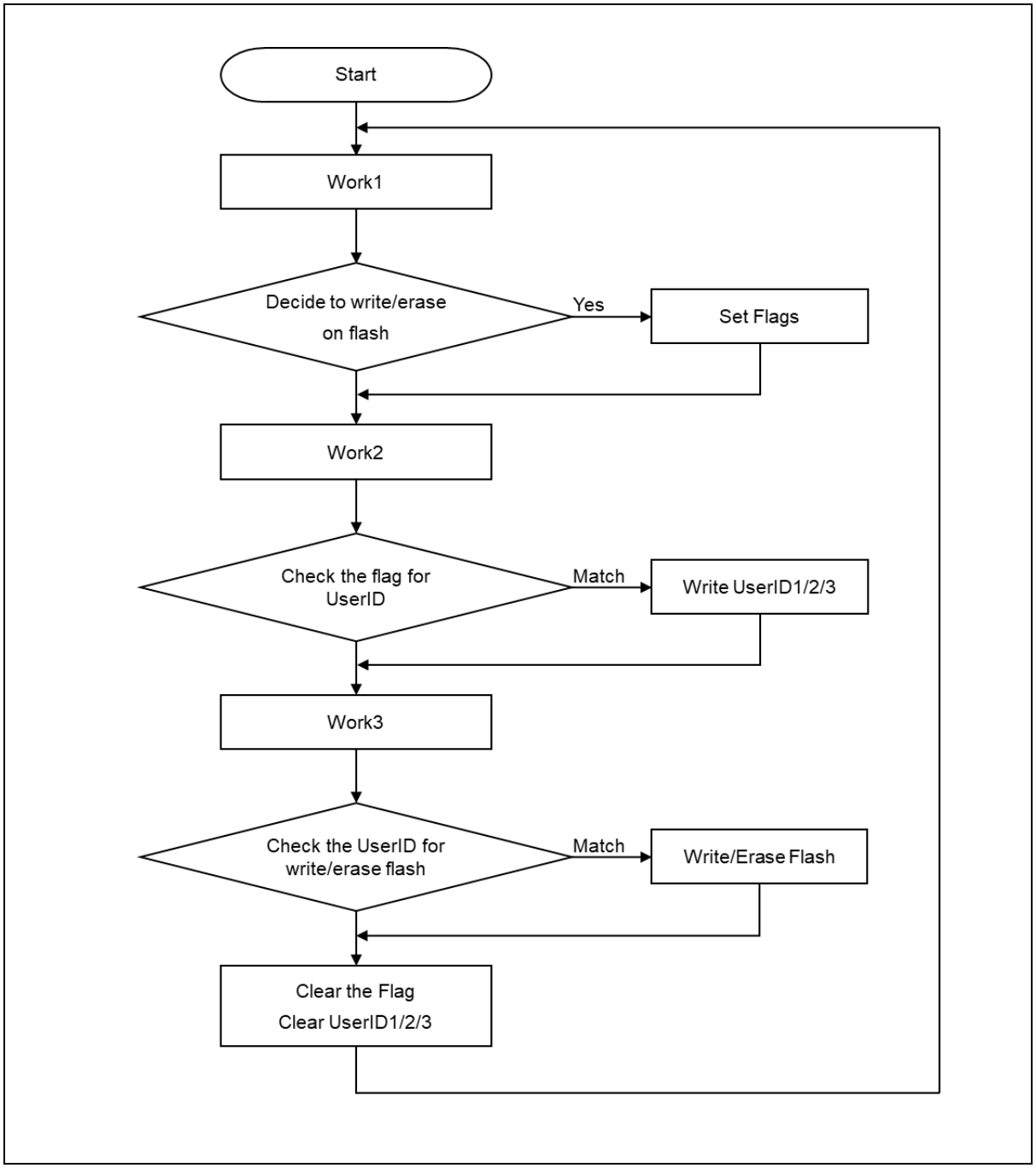
3. The following source code shows the overview of the main routine.

Figure 93. Overview of Main

```
1             ---
2             CALL     Work1
3             CALL     Decide_ErWt
4             CALL     Work2
5             CALL     ID_write
6             CALL     Work3
7             CALL     Flash_erase
8             CALL     Flash_write
9             ---
10            ---
11            ---
12
13   ID_wire:
14            MOV      A,#38H
15            CJNE     A,Flash_flag1,No_write_ID
16            MOV      A,#75H
17            CJNE     A,Flash_flag2,No_write_ID
18            MOV      UserID1,#ID_DATA_1          ;Write Uiser ID1
19            MOV      A,#38H
20            CJNE     A,Flash_flag1,No_write_ID
21            MOV      A,#75H
22            CJNE     A,Flash_flag2,No_write_ID
23            MOV      UserID2,#ID_DATA_2          ;Write Uiser ID2
24            MOV      A,#38H
25            CJNE     A,Flash_flag1,No_write_ID
26            MOV      A,#75H
27            CJNE     A,Flash_flag2,No_write_ID
28            MOV      UserID3,#ID_DATA_3          ;Write Uiser ID3
29            RET
30
31   No_write_ID:
32            MOV      UserID1,#00H
33            MOV      UserID2,#00H
34            MOV      UserID3,#00H
35            RET
```

20.8.1 Protection Flow of Invalid Erase/Write

Figure 94. Protection Flow of Invalid Erase/Write



20.9 Read Mode

The procedure for a Reading program in user program mode is as follows:

1. Load received data from flash memory on MOVC instruction by indirectly addressing mode.

Figure 95. Program Tip: Reading

```
1             MOV      A,#0
2             MOV      DPH,#0x0F
3             MOV      DPL,#0xA0          ;Flash memory address
4
5             MOVC     A,@A+DPTR          ;read data from Flash memory
```

20.10 Code Write Protection Mode

The code write-protection program procedure in user program mode:

1. Set Flash identification register (FIDR).
2. Check the UserID to prevent invalid operation [NOTE].
3. Set Flash mode control register (FMCR).

NOTE:

1. Refer to 20.8 Protection for Invalid Erase/Write.

Figure 96. Program Tip: Code Write Protection

```
1             MOV      FIDR,#0xA5         ;Identification value
2
3             MOV      A,#ID_DATA_1       ;Check the UserID(written by user)
4             CJNE     A,UserID1,No_WriteErase    ;This routine for UserID must be needed.
5             MOV      A,#ID_DATA_2
6             CJNE     A,UserID2,No_WriteErase
7
8             MOV      FMCR,#0x04         ;Start Flash Code Write Protection mode
9             NOP                         ;Dummy instruction, This instruction must be needed.
10            NOP                         ;Dummy instruction, This instruction must be needed.
11            NOP                         ;Dummy instruction, This instruction must be needed.
12
13   No_WriteErase:
14            MOV      FIDR,#00H
15            MOV      UserID1,#00H
16            MOV      UserID2,#00H
17            ---
```

21 Data Flash Memory

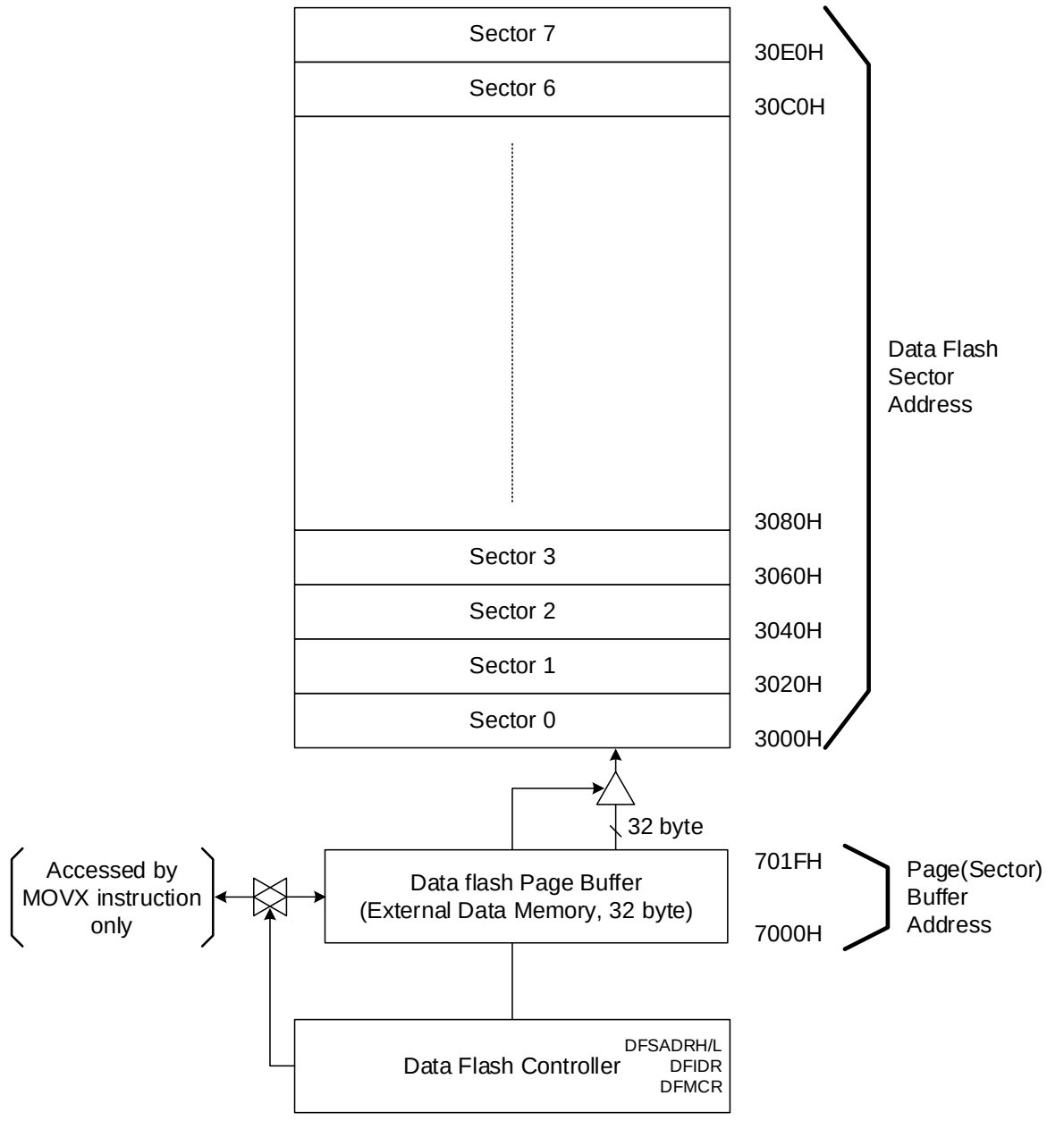
The A96L116 includes data flash memory of 256 bytes, which can be written, erased, and overwritten. The data flash memory can be read by ‘MOVX’ instruction.

- Data flash memory size: 256 bytes
- Single power supply program and erase
- Command interface for fast program and erase operation.
- Up to 100,000 program/erase cycles at typical voltage and temperature for memory

The write/erase cycles of the internal data flash memory can be increased significantly if it is divided into smaller and used in turn. If the 256 bytes are divided into eight areas with 32 bytes and each area from first to 8th is used up to 100,000 cycles, the total erase/write is for 800,000 cycles.

Figure 97 describes the relationship between data flash page buffer, data flash controller, and data flash sector addresses.

Figure 97. Data Flash Structure



NOTES:

1. The data flash memory doesn't support byte erase mode.
2. The data flash is in the extended data memory area.
3. If Idle and Stop are executed on data flash erase/write, the erase/write is forcibly terminated. So before entering Idle and Stop mode, the data flash busy bit (DFMBUSY) should be checked whether the bit is '1'. If the bit is '1', do not enter Idle and Stop mode.

21.1 Register Map

Table 37. Data Flash Register Map

Name	Address	Direction	Default	Description
DFSADRH	F3H	R/W	00H	Data Flash Sector Address High Register
DFSADRL	F2H	R/W	00H	Data Flash Sector Address Low Register
DFIDR	F4H	R/W	00H	Data Flash Identification Register
DFMCR	F5H	R/W	00H	Data Flash Mode Control Register

21.2 Register Description: Data Flash Control and Status

DFSADRH (Data Flash Sector Address High Register): F3H

7	6	5	4	3	2	1	0
DFSADRH7	DFSADRH6	DFSADRH5	DFSADRH4	DFSADRH3	DFSADRH2	DFSADRH1	DFSADRH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

DFSADRH[7:0] Data Flash Sector Address High

DFSADRL (Data Flash Sector Address Low Register): F2H

7	6	5	4	3	2	1	0
DFSADRL7	DFSADRL6	DFSADRL5	–	–	–	–	–
R/W	R/W	R/W	–	–	–	–	–

Initial value: 00H

DFSADRL[7:5] Data Flash Sector Address Low

DFIDR (Data Flash Identification Register): F4H

7	6	5	4	3	2	1	0
DFIDR7	DFIDR6	DFIDR5	DFIDR4	DFIDR3	DFIDR2	DFIDR1	DFIDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value: 00H

DFIDR[7:0] Data Flash Identification

Others No identification value

01101001 Identification value for a Data Flash mode

(These bits are automatically cleared to “00H” immediately after one time operation except “Data Flash page buffer reset mode”)

DFMCR (Data Flash Mode Control Register): F5H

7	6	5	4	3	2	1	0
DFMBUSY	–	–	–	–	DFMCR2	DFMCR1	DFMCR0
R	–	–	–	–	R/W	R/W	R/W

Initial value: 00H

DFMBUSY Data Flash busy bit.

0 No effect when ‘1’ is written

1 Busy

DFMCR[2:0] Data Flash Mode Control Bits

DFMCR2	DFMCR1	DFMCR0	Description
0	0	1	Select Data Flash page buffer reset mode and start regardless of the DFIDR value. (Clear all 16 bytes to “00H”)
0	1	0	Select Data Flash sector erase mode and start operation when the DFIDR[7:0]=”01101001”
1	0	0	Select Data Flash sector write mode and start operation when the DFIDR[7:0]=”01101001”
1	1	0	Select Data Flash bulk erase mode and start operation when the DFIDR[7:0]=”01101001”

Other Values: No operation

(Automatically cleared to “00H” immediately after one time operation)

21.3 Erase Mode

The sector erase program procedure in user program mode:

1. Page buffer clear (DFMCR=0x01)
2. Write '0' to page buffer.
3. Set Data Flash sector address register (DFSADRH/DFSADRL).
4. Set Data Flash identification register (DFIDR).
5. Check the UserID to prevent invalid operation [NOTE].
6. Set Data Flash mode control register (DFMCR).
7. Erase verify.

NOTE:

1. Refer to 20.8 Protection for Invalid Erase/Write.

Figure 98. Program Tip: Sector Erase

```
1             ANL      EO,#0xF8           ;Set DPTR0
2             MOV      DFMCR,#0x01        ;page buffer clear
3             NOP                         ;Dummy instruction, This instruction must be needed.
4             NOP                         ;Dummy instruction, This instruction must be needed.
5             NOP                         ;Dummy instruction, This instruction must be needed.
6
7             MOV      A,#0
8             MOV      R0,#DF_SectorSize  ;Sector size of Data Flash
9             MOV      DPH,#0x70          ;Page Buffer Address is 7000H
10            MOV      DPL,#0
11
12  DF_Pgbuf_clr:
13            MOVX     @DPTR,A
14            INC      DPTR
15            DJNZ     R0,DF_Pgbuf_clr    ;Write '0' to all page buffer
16
17            MOV      DFSADRH,#SAH       ;Sector Address High Byte.
18            MOV      DFSADRL,#SAL       ;Sector Address Low Byte
19            MOV      DFIDR,#0x69        ;Identification value
20
21            MOV      A,#DF_ID_DATA_1                    ;Check the UserID(written by user)
22            CJNE     A,DF_UserID1,No_DFWriteErase       ;This routine for UserID must be needed.
23            MOV      A,#DF_ID_DATA_2
24            CJNE     A,DF_UserID2,No_DFWriteErase
25
26            MOV      DFMCR,#0x02        ;Start Data Flash erase mode
27            NOP                         ;Dummy instruction, This instruction must be needed.
28            NOP                         ;Dummy instruction, This instruction must be needed.
29            NOP                         ;Dummy instruction, This instruction must be needed.
30
31            LJMP     DF_Erase_verify
32            ---
33  No_DFWriteErase:
34            MOV      DFIDR,#00H
35            MOV      DF_UserID1,#00H
36            MOV      DF_UserID2,#00H
37            ---
38  DF_Erase_verify:
39            MOV      A,DFMCR
40            JNB      ACC.7,DF_Erase_verify
41            ---
42  DF_Verify_error:
43            ---
```

21.4 Write Mode

The sector Write program procedure in user program mode.

1. Page buffer clear (DFMCR=0x01)
2. Write data to page buffer.
3. Set Data Flash sector address register (DFSADRH/DFSADRL).
4. Set Data Flash identification register (DFIDR).
5. Check the UserID to prevent invalid operation [NOTE].
6. Set Data Flash mode control register (DFMCR).
7. Write verify.

NOTE:

1. The data of the address must be “00H” before writing data to an address.

Figure 99. Program Tip: Sector Write

```
1             ANL       EO,#0xF8           ;Set DPTR0
2             MOV       DFMCR,#0x01        ;page buffer clear
3             NOP                          ;Dummy instruction, This instruction must be needed.
4             NOP                          ;Dummy instruction, This instruction must be needed.
5             NOP                          ;Dummy instruction, This instruction must be needed.
6
7             MOV       A,#0
8             MOV       R0,#DF_SectorSize  ;Sector size of DATA FLASH
9             MOV       DPH,#0x70          ;Page Buffer Address is 7000H
10            MOV       DPL,#0
11
12        DF_Pgbuf_WR:
13            MOVX      @DPTR,A
14            INC       A
15            INC       DPTR
16            DJNZ      R0,DF_Pgbuf_WR     ;Write data to all page buffer
17
18            MOV       DFSADRH,#SAH       ;Sector Address High Byte.
19            MOV       DFSADRL,#SAL       ;Sector Address Low Byte
20            MOV       DFIDR,#0x69        ;Identification value
21
22            MOV       A,#DF_ID_DATA_1                      ;Check the UserID(written by user)
23            CJNE      A,DF_UserID1,No_DFWriteErase         ;This routine for UserID must be needed.
24            MOV       A,#DF_ID_DATA_2
25            CJNE      A,DF_UserID2,No_DFWriteErase
26
27            MOV       DFMCR,#0x04        ;Start DATA FLASH write mode
28            NOP                          ;Dummy instruction, This instruction must be needed.
29            NOP                          ;Dummy instruction, This instruction must be needed.
30            NOP                          ;Dummy instruction, This instruction must be needed.
31
32            LJMP      DF_Write_verify
33            ---
34        No_DFWriteErase:
35            MOV       DFIDR,#00H
36            MOV       DF_UserID1,#00H
37            MOV       DF_UserID2,#00H
38            ---
39        DF_Write_verify:
40            MOV       A,DFMCR
41            JNB       ACC.7,DF_Write_verify
42            ---
43        DF_Verify_error:
44            ---
```

The Byte Write program procedure in user program mode.

1. Page buffer clear (DFMCR=0x01)
2. Write data to page buffer.
3. Set Data Flash sector address register (DFSADRH/DFSADRL).
4. Set Data Flash identification register (DFIDR).
5. Check the UserID to prevent invalid operation **NOTE**.
6. Set Data Flash mode control register (DFMCR).
7. Write verify.

NOTE:

1. The data of the address must be “00H” before writing data to an address.

Figure 100. Program Tip: Byte Write

```
1             ANL       EO,#0xF8           ;Set DPTR0
2             MOV       DFMCR,#0x01        ;page buffer clear
3             NOP                          ;Dummy instruction, This instruction must be needed.
4             NOP                          ;Dummy instruction, This instruction must be needed.
5             NOP                          ;Dummy instruction, This instruction must be needed.
6
7             MOV       A,#5
8             MOV       DPH,#0x70
9             MOV       DPL,#0
10            MOVX      @DPTR,A            ;Write data to page buffer
11
12            MOV       A,#6
13            MOV       DPH,#0x70
14            MOV       DPL,#0x05
15            MOVX      @DPTR,A            ;Write data to page buffer
16
17            MOV       DFSADRH,#SAH       ;Sector Address High Byte.
18            MOV       DFSADRL,#SAL       ;Sector Address Low Byte
19            MOV       DFIDR,#0x69        ;Identification value
20
21            MOV       A,#DF_ID_DATA_1                    ;Check the UserID(written by user)
22            CJNE      A,DF_UserID1,No_DFWriteErase       ;This routine for UserID must be needed.
23            MOV       A,#DF_ID_DATA_2
24            CJNE      A,DF_UserID2,No_DFWriteErase
25
26            MOV       DFMCR,#0x04        ;Start DATA FLASH write mode
27            NOP                          ;Dummy instruction, This instruction must be needed.
28            NOP                          ;Dummy instruction, This instruction must be needed.
29            NOP                          ;Dummy instruction, This instruction must be needed.
30
31            LJMP      DF_Write_verify
32            ---
33        No_DFWriteErase:
34            MOV       DFIDR,#00H
35            MOV       DF_UserID1,#00H
36            MOV       DF_UserID2,#00H
37            ---
38        DF_Write_verify:
              MOV       A,DFMCR
              JNB       ACC.7,DF_Write_verify
              ---
          DF_Verify_error:
              ---
```

21.5 Read Mode

The procedure for a Reading program in user program mode is as follows:

1. Load the received data from data flash memory on MOVX instruction by indirectly addressing mode.

Figure 101. Program Tip: Reading

```
1               MOV     DPH,#0x30
2               MOV     DPL,#0x10          ;Data Flash memory address
3
4               MOVX    A,@DPTR            ;read data from Data Flash memory
```

22 UNIQUE ID

22.1 Register Map

Table 38. UNIQUE ID Register

Name	Address	Direction	Default	Description
UNIQUEID0	1060H	R	xxH	Unique ID Register 0
UNIQUEID1	1061H	R	xxH	Unique ID Register 1
UNIQUEID2	1062H	R	xxH	Unique ID Register 2
UNIQUEID3	1063H	R	xxH	Unique ID Register 3
UNIQUEID4	1064H	R	xxH	Unique ID Register 4
UNIQUEID5	1065H	R	xxH	Unique ID Register 5
UNIQUEID6	1066H	R	xxH	Unique ID Register 6
UNIQUEID7	1067H	R	xxH	Unique ID Register 7
UNIQUEID8	1068H	R	xxH	Unique ID Register 8
UNIQUEID9	1069H	R	xxH	Unique ID Register 9
UNIQUEID10	106AH	R	xxH	Unique ID Register 10
UNIQUEID11	106BH	R	xxH	Unique ID Register 11
UNIQUEID12	106CH	R	xxH	Unique ID Register 12
UNIQUEID13	106DH	R	xxH	Unique ID Register 13
UNIQUEID14	106EH	R	xxH	Unique ID Register 14
UNIQUEID15	106FH	R	xxH	Unique ID Register 15

22.2 Register Description

UNIQUEID0 (Unique ID Register 0): 1060H (XSFR)

7	6	5	4	3	2	1	0
XYCDN31	XYCDN30	XYCDN29	XYCDN28	XYCDN27	XYCDN26	XYCDN25	XYCDN24
R	R	R	R	R	R	R	R

Initial value: xxH

XYCDN[31:24] X and Y Coordinates 1st byte

UNIQUEID1 (Unique ID Register 1): 1061H (XSFR)

7	6	5	4	3	2	1	0
XYCDN23	XYCDN22	XYCDN21	XYCDN20	XYCDN19	XYCDN18	XYCDN17	XYCDN16
R	R	R	R	R	R	R	R

Initial value: xxH

XYCDN[23:16] X and Y Coordinates 2nd byte

UNIQUEID2 (Unique ID Register 2): 1062H (XSFR)

7	6	5	4	3	2	1	0
XYCDN15	XYCDN14	XYCDN13	XYCDN12	XYCDN11	XYCDN10	XYCDN9	XYCDN8
R	R	R	R	R	R	R	R

Initial value: xxH

XYCDN[15:8] X and Y Coordinates 3rd byte

UNIQUEID3 (Unique ID Register 3): 1063H (XSFR)

7	6	5	4	3	2	1	0
XYCDN7	XYCDN6	XYCDN5	XYCDN4	XYCDN3	XYCDN2	XYCDN1	XYCDN0
R	R	R	R	R	R	R	R

Initial value: xxH

XYCDN[7:0] X and Y Coordinates 4th byte

UNIQUEID4 (Unique ID Register 4): 1064H (XSFR)

7	6	5	4	3	2	1	0
WAFNO7	WAFNO6	WAFNO5	WAFNO4	WAFNO3	WAFNO2	WAFNO1	WAFNO0
R	R	R	R	R	R	R	R

Initial value: xxH

WAFNO[7:0] Wafer Number

UNIQUEID5 (Unique ID Register 5): 1065H (XSFR)

7	6	5	4	3	2	1	0
LOTNO87	LOTNO86	LOTNO85	LOTNO84	LOTNO83	LOTNO82	LOTNO81	LOTNO80
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[87:80] Lot Number 1st byte

UNIQUEID6 (Unique ID Register 6): 1066H (XSFR)

7	6	5	4	3	2	1	0
LOTNO79	LOTNO78	LOTNO77	LOTNO76	LOTNO75	LOTNO74	LOTNO73	LOTNO72
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[79:72] Lot Number 2nd byte

UNIQUEID7 (Unique ID Register 7): 1067H (XSFR)

7	6	5	4	3	2	1	0
LOTNO71	LOTNO70	LOTNO69	LOTNO68	LOTNO67	LOTNO66	LOTNO65	LOTNO64
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[71:64] Lot Number 3rd byte

UNIQUEID8 (Unique ID Register 8): 1068H (XSFR)

7	6	5	4	3	2	1	0
LOTNO63	LOTNO62	LOTNO61	LOTNO60	LOTNO59	LOTNO58	LOTNO57	LOTNO56
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[63:56] Lot Number 4th byte

UNIQUEID9 (Unique ID Register 9): 1069H (XSFR)

7	6	5	4	3	2	1	0
LOTNO55	LOTNO54	LOTNO53	LOTNO52	LOTNO51	LOTNO50	LOTNO49	LOTNO48
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[55:48] Lot Number 5th byte

UNIQUEID10 (Unique ID Register 10): 106AH (XSFR)

7	6	5	4	3	2	1	0
LOTNO47	LOTNO46	LOTNO45	LOTNO44	LOTNO43	LOTNO42	LOTNO41	LOTNO40
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[47:40] Lot Number 6th byte

UNIQUEID11 (Unique ID Register 11): 106BH (XSFR)

7	6	5	4	3	2	1	0
LOTNO39	LOTNO38	LOTNO37	LOTNO36	LOTNO35	LOTNO34	LOTNO33	LOTNO32
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[39:32] Lot Number 7th byte

UNIQUEID12 (Unique ID Register 12): 106CH (XSFR)

7	6	5	4	3	2	1	0
LOTNO31	LOTNO30	LOTNO29	LOTNO28	LOTNO27	LOTNO26	LOTNO25	LOTNO24
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[31:24] Lot Number 8th byte

UNIQUEID13 (Unique ID Register 13): 106DH (XSFR)

7	6	5	4	3	2	1	0
LOTNO23	LOTNO22	LOTNO21	LOTNO20	LOTNO19	LOTNO18	LOTNO17	LOTNO16
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[23:16] Lot Number 9th byte

UNIQUEID14 (Unique ID Register 14): 106EH (XSFR)

7	6	5	4	3	2	1	0
LOTNO15	LOTNO14	LOTNO13	LOTNO12	LOTNO11	LOTNO10	LOTNO9	LOTNO8
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[15:8] Lot Number 10th byte

UNIQUEID15 (Unique ID Register 15): 106FH (XSFR)

7	6	5	4	3	2	1	0
LOTNO7	LOTNO6	LOTNO5	LOTNO4	LOTNO3	LOTNO2	LOTNO1	LOTNO0
R	R	R	R	R	R	R	R

Initial value: xxH

LOTNO[7:0] Lot Number 11th byte

Appendix A. Configure Option

A.1. Register Description: Configure Option Control

CONFIGURE OPTION 1: ROM Address 001FH

7	6	5	4	3	2	1	0
R_P	HL	–	VAPEN	–	–	–	RSTS

Initial value: 00H

R_P	Code Read Protection	
	0	Disable
	1	Enable
HL	Code Write Protection	
	0	Disable
	1	Enable
VAPEN	Vector Area (00H – FFH) Write Protection	
	0	Disable Protection (Erasable by instruction)
	1	Enable Protection (Not erasable by instruction)
RSTS	Select RESETB pin	
	0	Disable RESETB pin (P10)
	1	Enable RESETB pin

CONFIGURE OPTION 2: ROM Address 001EH

7	6	5	4	3	2	1	0
–	–	–	–	PAEN	PASS2	PASS1	PASS0

Initial value: 00H

PAEN	Enable Specific Area Write Protection	
	0	Disable (erasable by instruction)
	1	Enable (not erasable by instruction)
PASS [2:0]	Select Specific Area for Write Protection	

NOTE:

1. The PASS value is applied only when PAEN=1.

PASS2	PASS1	PASS0	
0	0	0	0.7 Kbytes (Address 0100H – 03FFH)
0	0	1	1.7 Kbytes (Address 0100H – 07FFH)
0	1	0	2.7 Kbytes (Address 0100H – 0BFFH)
0	1	1	3.7 KBytes (Address 0100H – 0FFFH)
1	0	0	13.7 Kbytes (Address 0100H – 37FFH)
1	0	1	14.7 Kbytes (Address 0100H – 3BFFH)
1	1	0	15.2 Kbytes (Address 0100H – 3DFFH)
1	1	1	15.5 KBytes (Address 0100H – 3EFFH)

Appendix B. Instruction Table

- Instructions are either one, two or three bytes long as listed in the ‘Bytes’ column in tables shown below.
- Each instruction takes either one, two or four machine cycles to execute as listed in the following tables in this section.
- One machine cycle comprises two system clock cycles.

Table 39. Instruction Table: Arithmetic

Arithmetic				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
ADD A,Rn	Add register to A	1	1	28-2F
ADD A,dir	Add direct byte to A	2	1	25
ADD A,@Ri	Add indirect memory to A	1	1	26-27
ADD A,#data	Add immediate to A	2	1	24
ADDC A,Rn	Add register to A with carry	1	1	38-3F
ADDC A,dir	Add direct byte to A with carry	2	1	35
ADDC A,@Ri	Add indirect memory to A with carry	1	1	36-37
ADDC A,#data	Add immediate to A with carry	2	1	34
SUBB A,Rn	Subtract register from A with borrow	1	1	98-9F
SUBB A,dir	Subtract direct byte from A with borrow	2	1	95
SUBB A,@Ri	Subtract indirect memory from A with borrow	1	1	96-97
SUBB A,#data	Subtract immediate from A with borrow	2	1	94
INC A	Increment A	1	1	04
INC Rn	Increment register	1	1	08-0F
INC dir	Increment direct byte	2	1	05
INC @Ri	Increment indirect memory	1	1	06-07
DEC A	Decrement A	1	1	14
DEC Rn	Decrement register	1	1	18-1F
DEC dir	Decrement direct byte	2	1	15
DEC @Ri	Decrement indirect memory	1	1	16-17
INC DPTR	Increment data pointer	1	2	A3
MUL AB	Multiply A by B	1	4	A4
DIV AB	Divide A by B	1	4	84
DA A	Decimal Adjust A	1	1	D4

Table 40. Instruction Table: Logical

Logical				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
ANL A,Rn	AND register to A	1	1	58-5F
ANL A,dir	AND direct byte to A	2	1	55
ANL A,@Ri	AND indirect memory to A	1	1	56-57
ANL A,#data	AND immediate to A	2	1	54
ANL dir,A	AND A to direct byte	2	1	52
ANL dir,#data	AND immediate to direct byte	3	2	53
ORL A,Rn	OR register to A	1	1	48-4F
ORL A,dir	OR direct byte to A	2	1	45
ORL A,@Ri	OR indirect memory to A	1	1	46-47
ORL A,#data	OR immediate to A	2	1	44
ORL dir,A	OR A to direct byte	2	1	42
ORL dir,#data	OR immediate to direct byte	3	2	43
XRL A,Rn	Exclusive-OR register to A	1	1	68-6F
XRL A,dir	Exclusive-OR direct byte to A	2	1	65
XRL A, @Ri	Exclusive-OR indirect memory to A	1	1	66-67
XRL A,#data	Exclusive-OR immediate to A	2	1	64
XRL dir,A	Exclusive-OR A to direct byte	2	1	62
XRL dir,#data	Exclusive-OR immediate to direct byte	3	2	63
CLR A	Clear A	1	1	E4
CPL A	Complement A	1	1	F4
SWAP A	Swap Nibbles of A	1	1	C4
RL A	Rotate A left	1	1	23
RLC A	Rotate A left through carry	1	1	33
RR A	Rotate A right	1	1	03
RRC A	Rotate A right through carry	1	1	13

Table 41. Instruction Table: Data Transfer

Data transfer				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
MOV A,Rn	Move register to A	1	1	E8-EF
MOV A,dir	Move direct byte to A	2	1	E5
MOV A,@Ri	Move indirect memory to A	1	1	E6-E7
MOV A,#data	Move immediate to A	2	1	74
MOV Rn,A	Move A to register	1	1	F8-FF
MOV Rn,dir	Move direct byte to register	2	2	A8-AF
MOV Rn,#data	Move immediate to register	2	1	78-7F
MOV dir,A	Move A to direct byte	2	1	F5
MOV dir,Rn	Move register to direct byte	2	2	88-8F
MOV dir,dir	Move direct byte to direct byte	3	2	85
MOV dir,@Ri	Move indirect memory to direct byte	2	2	86-87
MOV dir,#data	Move immediate to direct byte	3	2	75
MOV @Ri,A	Move A to indirect memory	1	1	F6-F7
MOV @Ri,dir	Move direct byte to indirect memory	2	2	A6-A7
MOV @Ri,#data	Move immediate to indirect memory	2	1	76-77
MOV DPTR,#data	Move immediate to data pointer	3	2	90
MOVC A,@A+DPTR	Move code byte relative DPTR to A	1	2	93
MOVC A,@A+PC	Move code byte relative PC to A	1	2	83
MOVX A,@Ri	Move external data(A8) to A	1	2	E2-E3
MOVX A,@DPTR	Move external data(A16) to A	1	2	E0
MOVX @Ri,A	Move A to external data(A8)	1	2	F2-F3
MOVX @DPTR,A	Move A to external data(A16)	1	2	F0
PUSH dir	Push direct byte onto stack	2	2	C0
POP dir	Pop direct byte from stack	2	2	D0
XCH A,Rn	Exchange A and register	1	1	C8-CF
XCH A,dir	Exchange A and direct byte	2	1	C5
XCH A,@Ri	Exchange A and indirect memory	1	1	C6-C7
XCHD A,@Ri	Exchange A and indirect memory nibble	1	1	D6-D7

Table 42. Instruction Table: Boolean

Boolean				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
CLR C	Clear carry	1	1	C3
CLR bit	Clear direct bit	2	1	C2
SETB C	Set carry	1	1	D3
SETB bit	Set direct bit	2	1	D2
CPL C	Complement carry	1	1	B3
CPL bit	Complement direct bit	2	1	B2
ANL C,bit	AND direct bit to carry	2	2	82
ANL C,/bit	AND direct bit inverse to carry	2	2	B0
ORL C,bit	OR direct bit to carry	2	2	72
ORL C,/bit	OR direct bit inverse to carry	2	2	A0
MOV C,bit	Move direct bit to carry	2	1	A2
MOV bit,C	Move carry to direct bit	2	2	92

Table 43. Instruction Table: Branching

Branching				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
ACALL addr 11	Absolute jump to subroutine	2	2	11→F1
LCALL addr 16	Long jump to subroutine	3	2	12
RET	Return from subroutine	1	2	22
RETI	Return from interrupt	1	2	32
AJMP addr 11	Absolute jump unconditional	2	2	01→E1
LJMP addr 16	Long jump unconditional	3	2	02
SJMP rel	Short jump (relative address)	2	2	80
JC rel	Jump on carry=1	2	2	40
JNC rel	Jump on carry=0	2	2	50
JB bit,rel	Jump on direct bit=1	3	2	20
JNB bit,rel	Jump on direct bit=0	3	2	30
JBC bit,rel	Jump on direct bit=1 and clear	3	2	10
JMP @A+DPTR	Jump indirect relative DPTR	1	2	73
JZ rel	Jump on accumulator=0	2	2	60
JNZ rel	Jump on accumulator ≠0	2	2	70
CJNE A,dir,rel	Compare A,direct jne relative	3	2	B5
CJNE A,#d,rel	Compare A,immediate jne relative	3	2	B4
CJNE Rn,#d,rel	Compare register, immediate jne relative	3	2	B8-BF
CJNE @Ri,#d,rel	Compare indirect, immediate jne relative	3	2	B6-B7
DJNZ Rn,rel	Decrement register, jnz relative	2	2	D8-DF
DJNZ dir,rel	Decrement direct byte, jnz relative	3	2	D5

Table 44. Instruction Table: Miscellaneous

Miscellaneous				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
NOP	No operation	1	1	00

Table 45. Instruction Table: Additional Instructions

Additional instructions (selected through EO[7:4])				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
MOVC @(DPTR++),A	M8051W/M8051EW-specific instruction supporting software download into program memory	1	2	A5
TRAP	Software break command	1	1	A5

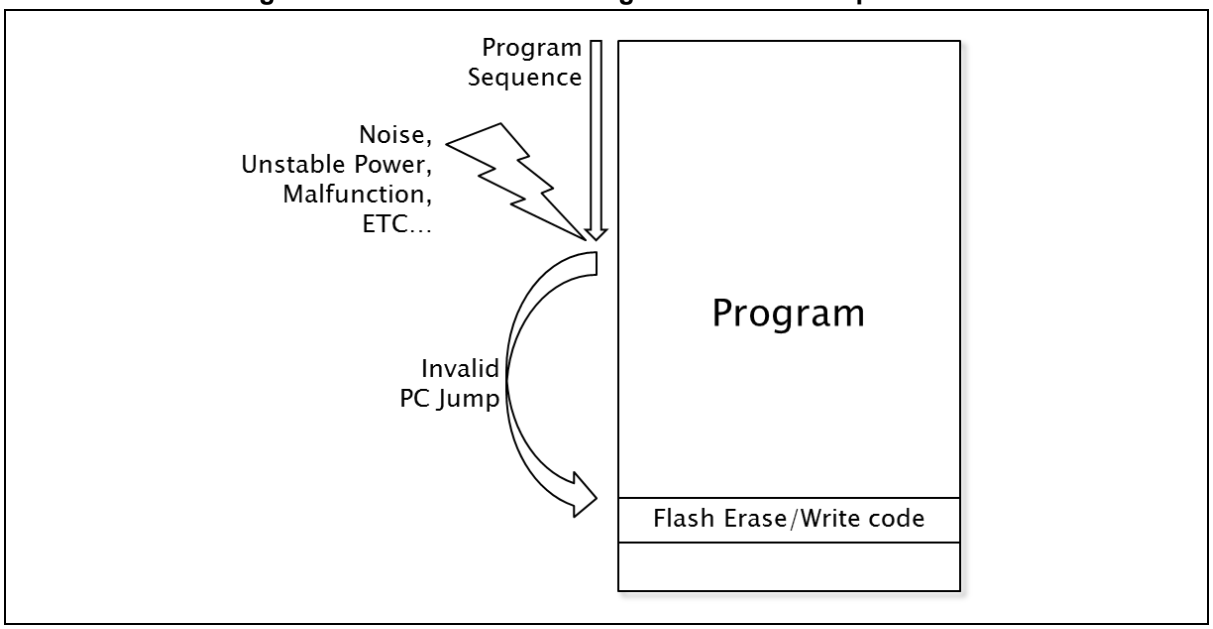
In the above table, an entry such as E8-EF indicates a continuous block of hex opcodes used for 8 different registers, and the register numbers of which are defined by the lowest three bits of the corresponding code. Non-continuous blocks of codes, shown as 11→F1 (for example), are used for absolute jumps and calls, with the top three bits of the code being used to store the top three bits of the destination address.

The CJNE instructions use the abbreviation #d for immediate data; other instructions use #data.

Appendix C. Flash Protection for Invalid Erase/Write

Appendix C shows example code to prevent code or data from being changed by abnormal operations such as noise, unstable power, and malfunction.

Figure 102. Flash Protection Against Abnormal Operations



C.1. How to Protect Flash

- Divide into decision and execution to Erase/Write in flash.
  - Check the program sequence from decision to execution in order of precedence about Erase/Write.
  - Setting the flags in program and check the flags in main loop at the end.
  - When the Flash Erase/Write is executed, check the flags. If not matched, do not execute.
- Check the range of Flash Sector Address
  - If the Flash sector address is outside of specific area, do not execute.
- Use the Dummy Address
  - Set the Flash sector address to dummy address in usually run time.
  - Change the Flash sector address to real area range shortly before Erase/Write.
  - Even if invalid Erase/Write occurred, it will be Erase/Write in dummy address in flash.
- Use the LVR/LVI
  - Unstable or low powers give an adverse effect on microcontroller. So, use the LVR/LVI

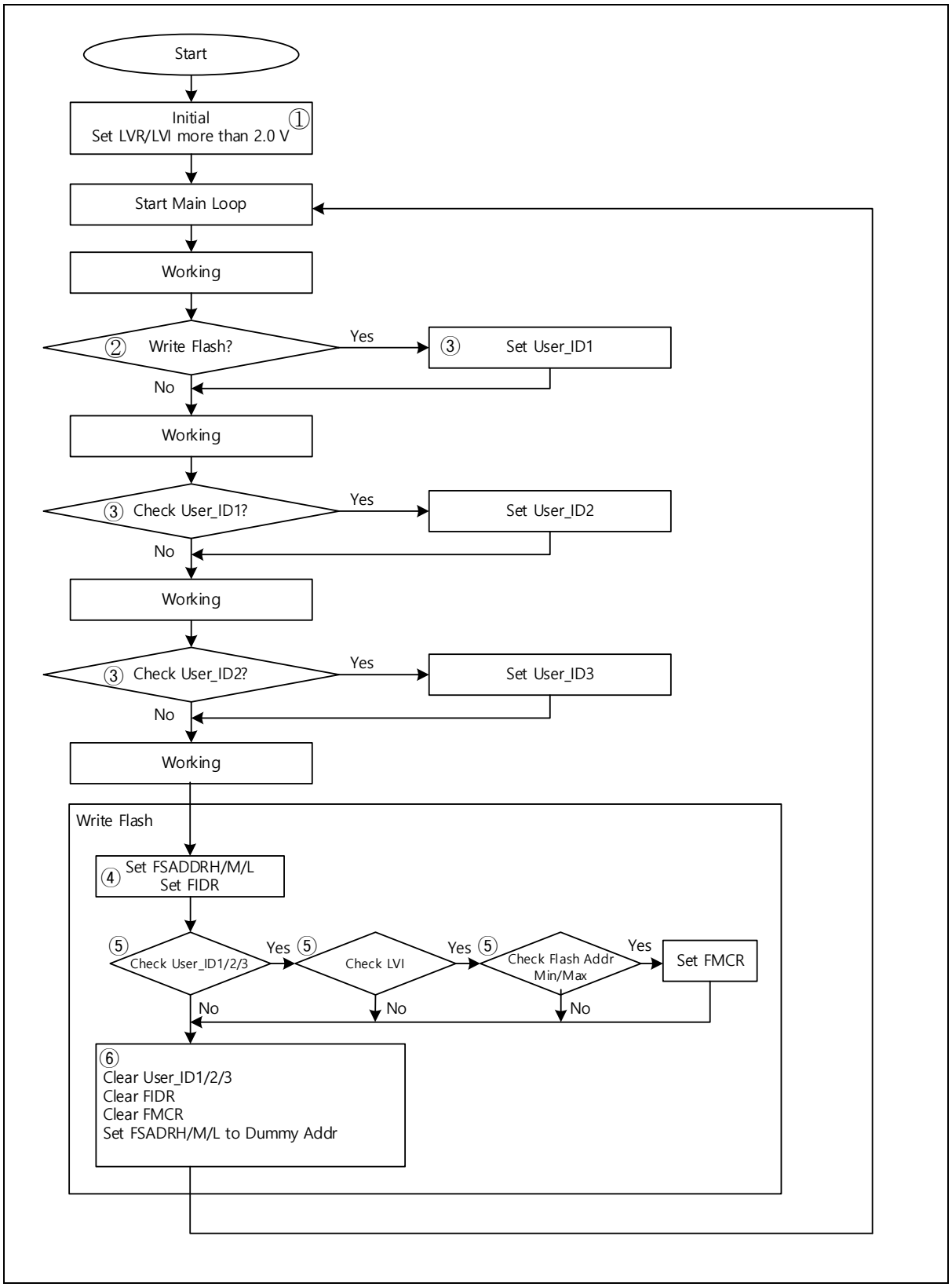
C.2. Protection Flow Description

The Flash protection procedure is described in the flowchart in Figure 103, and each step in this figure is described in the following lists:

1. Initialization
   A. Set the LVR/LVI. Check the power by LVR/LVI and do not execute under unstable or low power.
   B. Initialize User_ID1/2/3
   C. Set Flash Sector Address High/Middle/Low to dummy address. Dummy address is set to unused area range in flash.
2. Decide to Write
   A. When the Erase/Write are determined, set flag. Do not directly Erase/Write in flash.
   B. Make the user data.
3. Check and Set User_ID1/2/3
   A. In the middle of source, insert code which can check and set the flags.
   B. By setting the User_ID 1/2/3 sequentially, identify the flow of the program.
4. Set Flash Sector Address
   A. Set address to real area range shortly before Erase/Write in flash.
   B. Set to dummy address after Erase/Write. Even if invalid work occurred, it will be Erase/Write in dummy address in the flash memory.
5. Check Flags
   A. If every flag (User_ID1/2/3, LVI, Flash Address Min/Max) was set, then do Erase/Write.
   B. If the Flash Sector Address is outside of Min/Max, do not execute.
   C. Address Min/Max is set to unused area.
6. Initialize Flags
   A. Initialize User_ID1/2/3
   B. Set Flash Sector Address to Dummy Address

- Sample Source
  - Refer to the ABOV website (https://www.abovsemi.com/).
  - It is created based on the MC97F2664.
  - Each product should be modified according to the Page Buffer Size and Flash Memory Size

Figure 103. Flowchart of Flash Protection



C.3. Other Protection by Configure Options

- Protection by Configure Options:
  - Set Flash protection by microcontroller writing tools (OCD, PGM+, etc.)
    - Vector Area:

      00H~FFH
    - Specific Area (A96L116):

      0.7 KBytes (Address 0100H – 03FFH)

      1.7 KBytes (Address 0100H – 07FFH)

      2.7 KBytes (Address 0100H – 0BFFH)

      3.8 KBytes (Address 0100H – 0FFFH)

      13.7 KBytes (Address 0100H – 37FFH)

      14.7 KBytes (Address 0100H – 3BFFH)

      15.2 KBytes (Address 0100H – 3DFFH)

      15.5 KBytes (Address 0100H – 3EFFH)
  - The range of protection may be different for each product.

Revision History

Revision	Date	Notes
1.00	Jan. 8, 2024	Initial release
1.01	Jan. 25, 2024	Corrected typos and made minor changes.
1.02	Apr. 24, 2024	Corrected typos and revised 20-QFN package dimensions.

Korea

Regional Office, Seoul
R&D, Marketing & Sales
8th Fl., 330, Yeongdong-daero,
Gangnam-gu, Seoul,
06177, Korea

Tel: +82-2-2193-2200
Fax: +82-2-508-6903
www.abovsemi.com

HQ, Ochang
R&D, QA, and Test Center
37, Gangni 1-gil, Ochang-eup,
Cheongwon-gun,
Chungcheongbuk-do,28126, Korea

Tel: +82-43-219-5200
Fax: +82-43-217-3534
www.abovsemi.com

Domestic Sales Manager
Tel: +82-2-2193-2206
Fax: +82-2-508-6903
Email: sales_kr@abov.co.kr

Global Sales Manager
Tel: +82-2-2193-2281
Fax: +82-2-508-6903
Email: sales_gl@abov.co.kr

China Sales Manager
Tel: +86-755-8287-2205
Fax: +86-755-8287-2204
Email: sales_cn@abov.co.kr

ABOV Disclaimer

IMPORTANT NOTICE – PLEASE READ CAREFULLY

ABOV Semiconductor ("ABOV") reserves the right to make changes, corrections, enhancements, modifications, and improvements to ABOV products and/or to this document at any time without notice. ABOV does not give warranties as to the accuracy or completeness of the information included herein. Purchasers should obtain the latest relevant information of ABOV products before placing orders. Purchasers are entirely responsible for the choice, selection, and use of ABOV products and ABOV assumes no liability for application assistance or the design of purchasers' products. No license, express or implied, to any intellectual property rights is granted by ABOV herein. ABOV disclaims all express and implied warranties and shall not be responsible or liable for any injuries or damages related to use of ABOV products in such unauthorized applications. ABOV and the ABOV logo are trademarks of ABOV. All other product or service names are the property of their respective owners. Information in this document supersedes and replaces the information previously supplied in any former versions of this document.

© 2024 ABOV Semiconductor – All rights reserved